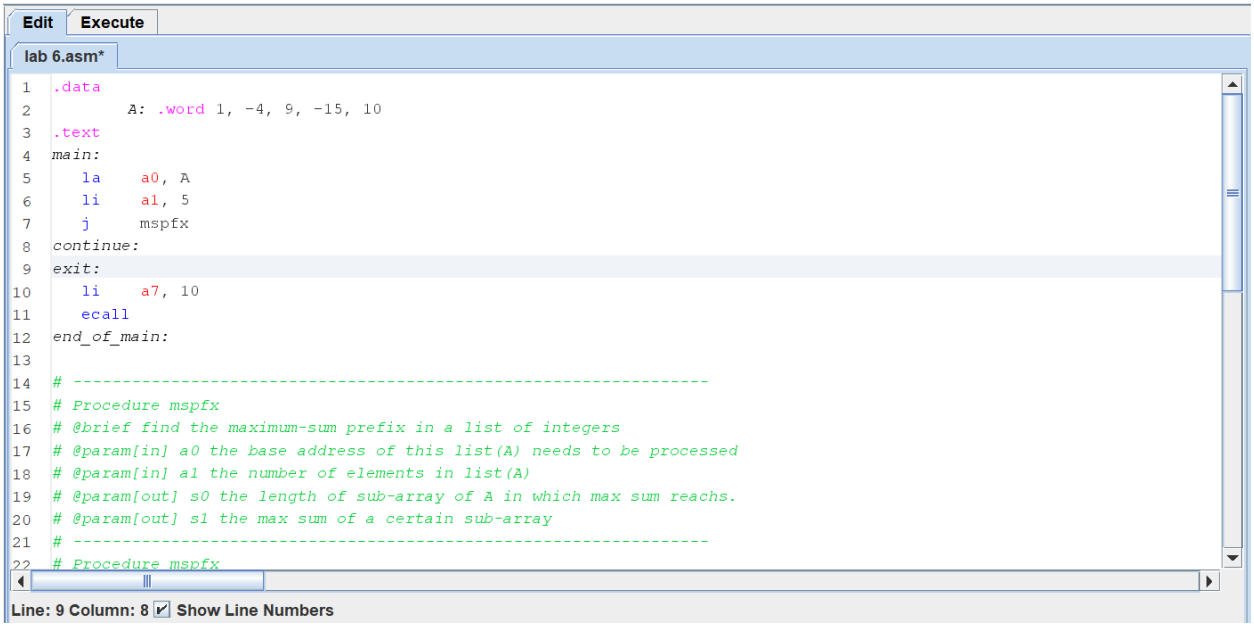
Lab 6:

Student name: Lê Ngọc Anh Vũ

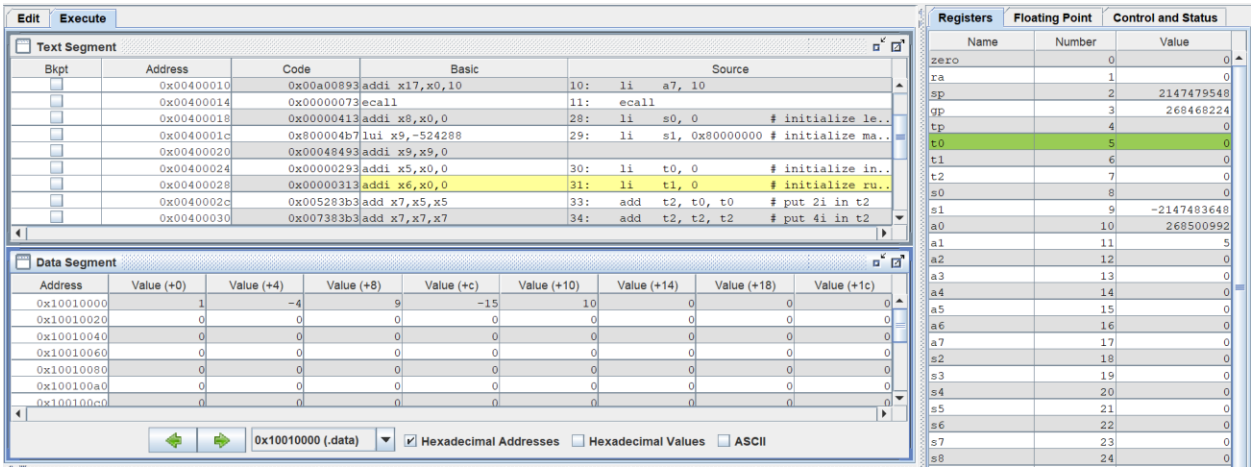
Student ID: 20236014

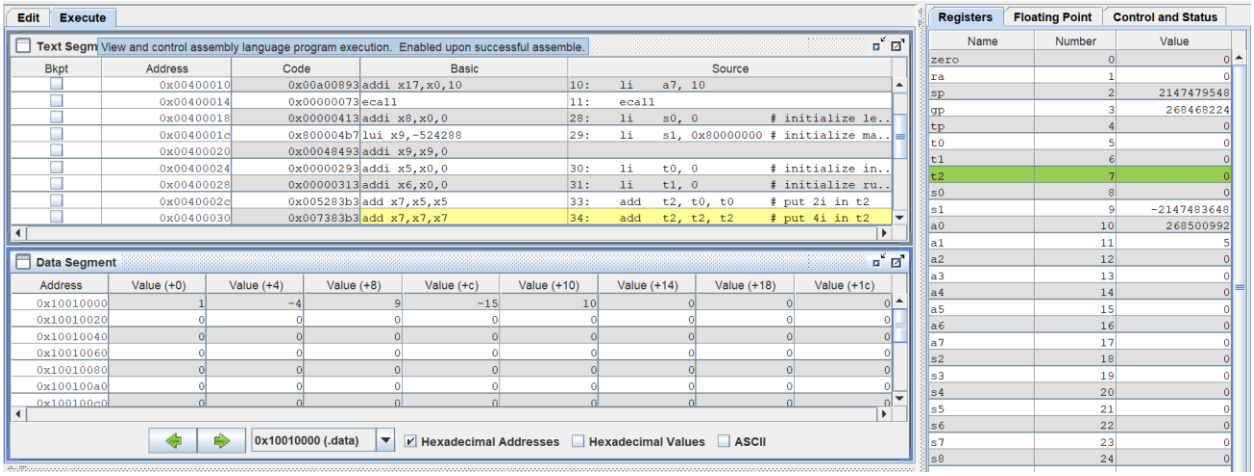
Assignment 1: Create a project that implements the program in Home Assignment 1. Initialize a new set of values for the array and compile it. Run the program step by step and observe the changes in the registers to verify that the program works according to the algorithm.

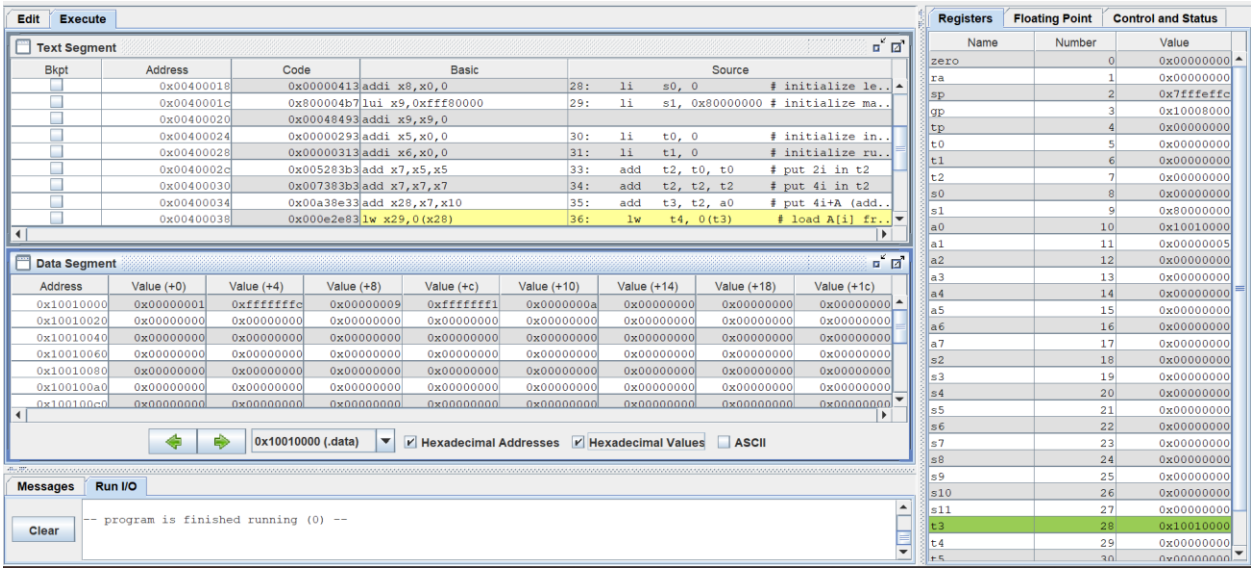
We initialize a new set {1, -4, 9, -15, 10} to A:

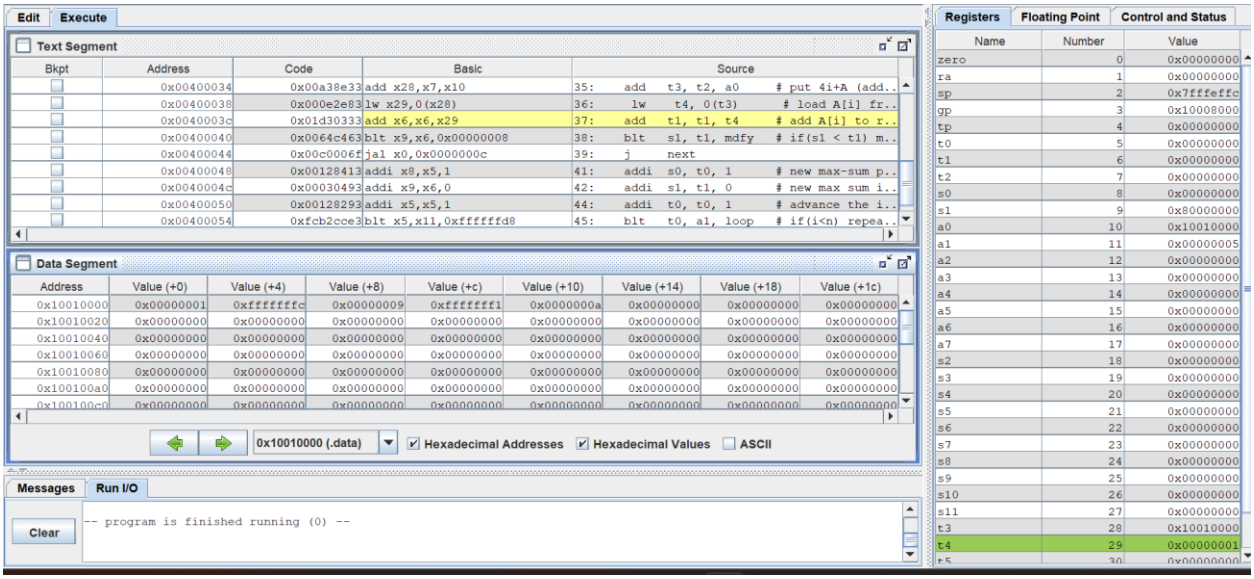


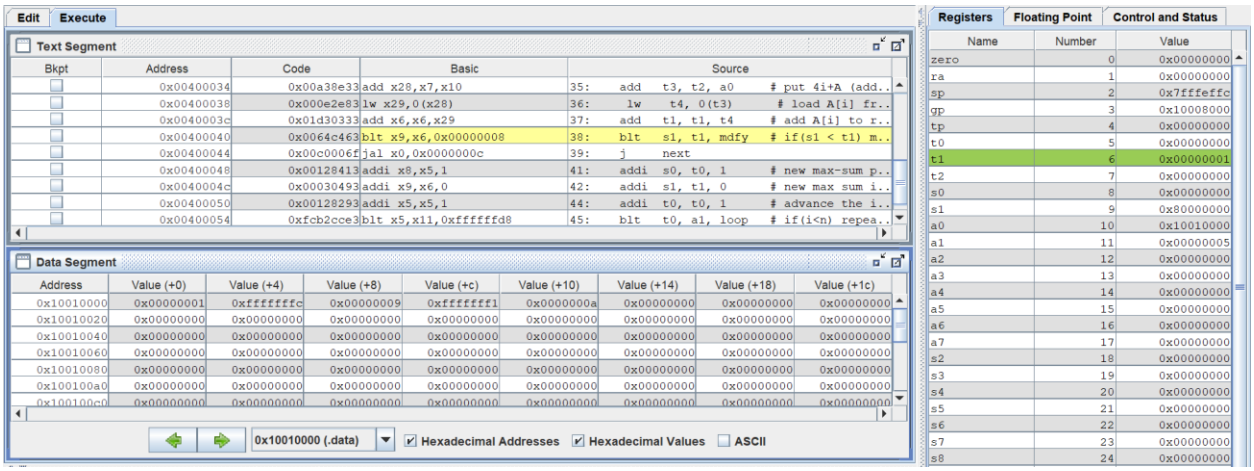
In the first iteration, i=t0=0, t2=0 and the register t3 gets the address of A[0], which is 0x10010000. Then, t4=A[0]=1. The running sum value in t1 is equal to 1 (A[0]). Because s1<t1, the program jumps to label mdfy. Hence, s1 has new value of 1.

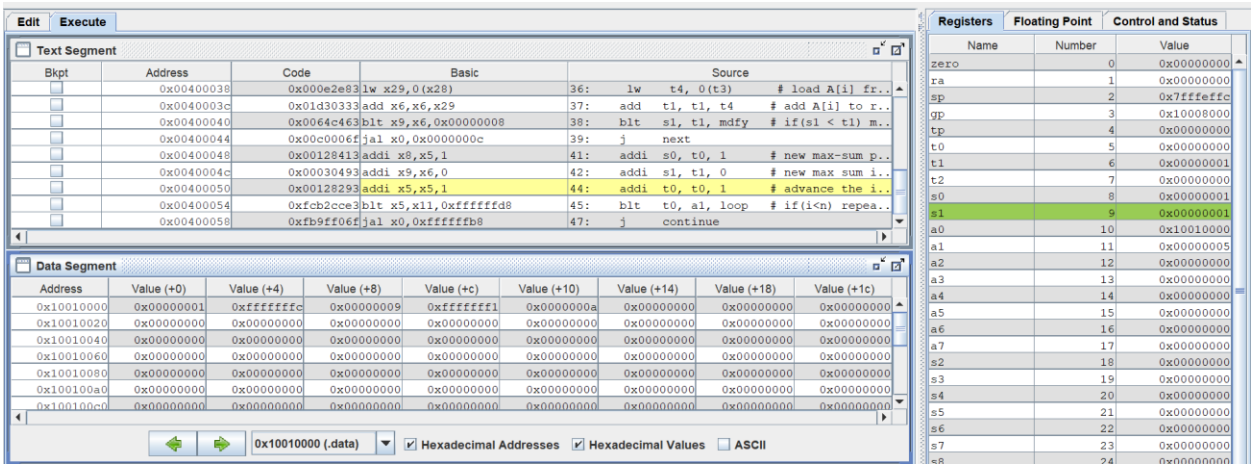



Edit
Execute
Text Segment
View and control assembly language program execution. Enabled upon successful assemble.
Bkpt
Address
Code
Basic
Source
0x00400010 0x00a00893 addi x17,x0,10 10: li a7, 10
0x00400014 0x00000073 ecall 11: ecall
0x00400018 0x00000413 addi x8,x0,0 28: li s0, 0 # initialize le..
0x0040001c 0x800004b7 lui x9,-524288 29: li s1, 0x80000000 # initialize ma..
0x00400020 0x00048493 addi x9,x9,0
0x00400024 0x00000293 addi x5,x0,0 30: li t0, 0 # initialize in..
0x00400028 0x00000313 addi x6,x0,0 31: li t1, 0 # initialize ru..
0x0040002c 0x005283b3 add x7,x5,x5 33: add t2, t0, t0 # put 2i in t2
0x00400030 0x007383b3 add x7,x7,x7 34: add t2, t2, t2 # put 4i in t2
Data Segment
Address
Value (+0)
Value (+4)
Value (+8)
Value (+c)
Value (+10)
Value (+14)
Value (+18)
Value (+1c)
0x10010000 1 -4 9 -15 10 0 0 0
0x10010020 0 0 0 0 0 0 0 0
0x10010040 0 0 0 0 0 0 0 0
0x10010060 0 0 0 0 0 0 0 0
0x10010080 0 0 0 0 0 0 0 0
0x100100a0 0 0 0 0 0 0 0 0
0x100100c0 0 0 0 0 0 0 0 0
0x10010000 (.data)
Hexadecimal Addresses
Hexadecimal Values
ASCII
Registers
Floating Point
Control and Status
Name
Number
Value
zero 0 0
ra 1 0
sp 2 2147479548
gp 3 268468224
tp 4 0
t0 5 0
t1 6 0
t2 7 0
s0 8 0
s1 9 -2147483648
a0 10 268500992
a1 11 5
a2 12 0
a3 13 0
a4 14 0
a5 15 0
a6 16 0
a7 17 0
s2 18 0
s3 19 0
s4 20 0
s5 21 0
s6 22 0
s7 23 0
s8 24 0

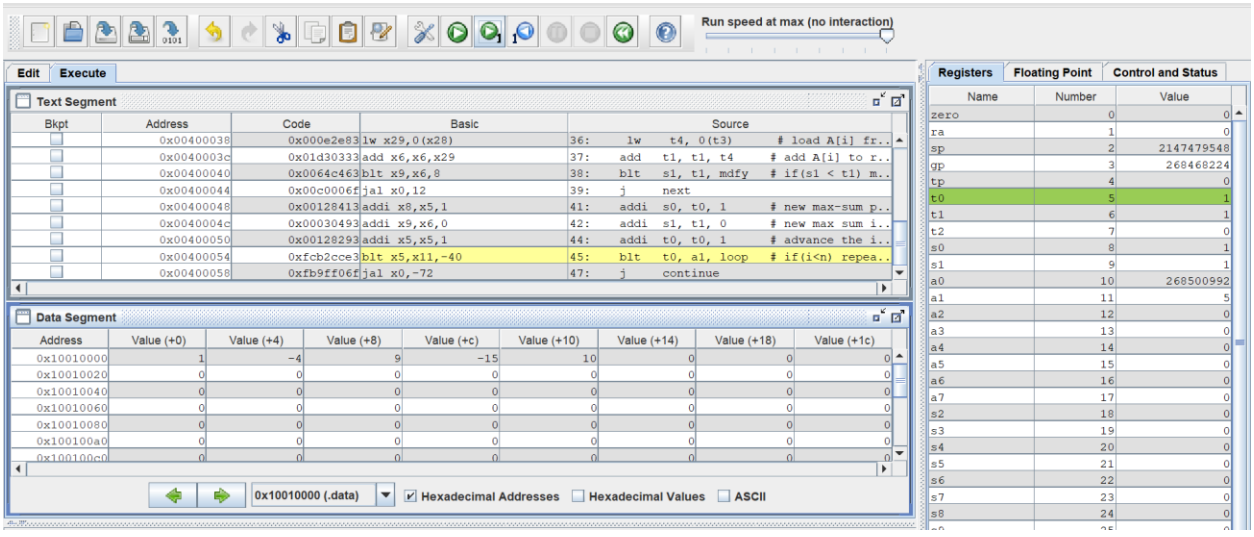

Edit
Execute
Text Segment
Bkpt
Address
Code
Basic
Source
0x00400018 0x00000413 addi x8,x0,0 28: li s0, 0 # initialize le..
0x0040001c 0x800004b7 lui x9,0xfff80000 29: li s1, 0x80000000 # initialize ma..
0x00400020 0x00048493 addi x9,x9,0
0x00400024 0x00000293 addi x5,x0,0 30: li t0, 0 # initialize in..
0x00400028 0x00000313 addi x6,x0,0 31: li t1, 0 # initialize ru..
0x0040002c 0x005283b3 add x7,x5,x5 33: add t2, t0, t0 # put 2i in t2
0x00400030 0x007383b3 add x7,x7,x7 34: add t2, t2, t2 # put 4i in t2
0x00400034 0x00a38e33 add x28,x7,x10 35: add t3, t2, a0 # put 4i+A (add..
0x00400038 0x000e2e83 lw x29,0(x28) 36: lw t4, 0(t3) # load A[i] fr..
Data Segment
Address
Value (+0)
Value (+4)
Value (+8)
Value (+c)
Value (+10)
Value (+14)
Value (+18)
Value (+1c)
0x10010000 0x00000001 0xfffffffc 0x00000009 0xfffffff1 0x0000000a 0x00000000 0x00000000 0x00000000
0x10010020 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x10010040 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x10010060 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x10010080 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x100100a0 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x100100c0 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x10010000 (.data)
Hexadecimal Addresses
Hexadecimal Values
ASCII
Messages
Run I/O
Clear
-- program is finished running (0) --
Registers
Floating Point
Control and Status
Name
Number
Value
zero 0 0x00000000
ra 1 0x00000000
sp 2 0x7fffeffc
gp 3 0x10008000
tp 4 0x00000000
t0 5 0x00000000
t1 6 0x00000000
t2 7 0x00000000
s0 8 0x00000000
s1 9 0x80000000
a0 10 0x10010000
a1 11 0x00000005
a2 12 0x00000000
a3 13 0x00000000
a4 14 0x00000000
a5 15 0x00000000
a6 16 0x00000000
a7 17 0x00000000
s2 18 0x00000000
s3 19 0x00000000
s4 20 0x00000000
s5 21 0x00000000
s6 22 0x00000000
s7 23 0x00000000
s8 24 0x00000000
s9 25 0x00000000
s10 26 0x00000000
s11 27 0x00000000
t3 28 0x10010000
t4 29 0x00000000

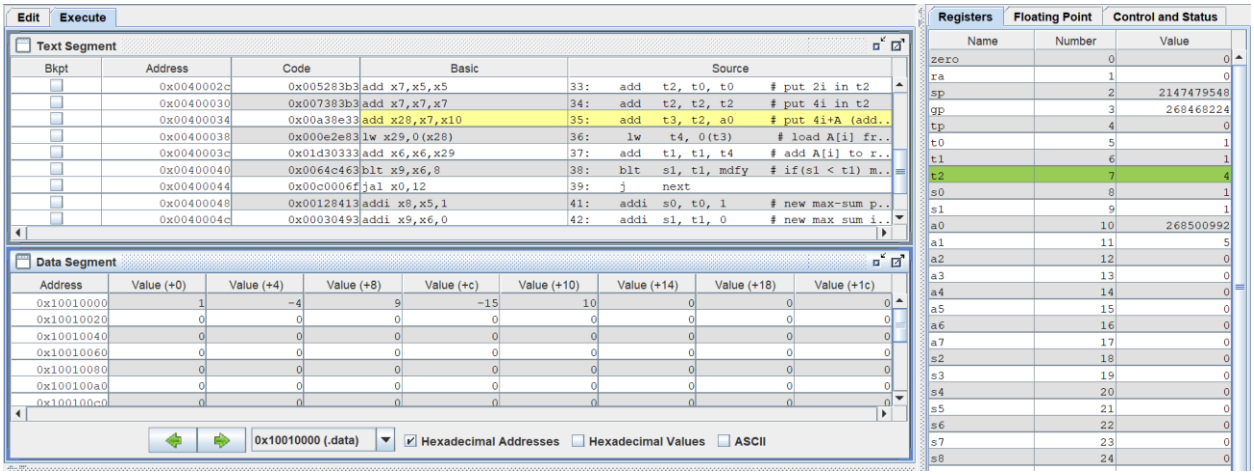

Edit
Execute
Text Segment
Bkpt
Address
Code
Basic
Source
0x00400034 0x00a38e33 add x28,x7,x10 35: add t3, t2, a0 # put 4i+A (add..
0x00400038 0x000e2e83 lw x29,0(x28) 36: lw t4, 0(t3) # load A[i] fr..
0x0040003c 0x01d30333 add x6,x6,x29 37: add t1, t1, t4 # add A[i] to r..
0x00400040 0x0064c463 blt x9,x6,0x00000008 38: blt s1, t1, mdfy # if(s1 < t1) m..
0x00400044 0x00c0006f jal x0,0x0000000c 39: j next
0x00400048 0x00128413 addi x8,x5,1 41: addi s0, t0, 1 # new max-sum p..
0x0040004c 0x00030493 addi x9,x6,0 42: addi s1, t1, 0 # new max sum i..
0x00400050 0x00128293 addi x5,x5,1 44: addi t0, t0, 1 # advance the i..
0x00400054 0xfcb2cce3 blt x5,x11,0xfffffd8 45: blt t0, a1, loop # if(i<n) repea..
Data Segment
Address
Value (+0)
Value (+4)
Value (+8)
Value (+c)
Value (+10)
Value (+14)
Value (+18)
Value (+1c)
0x10010000 0x00000001 0xfffffffc 0x00000009 0xfffffff1 0x0000000a 0x00000000 0x00000000 0x00000000
0x10010020 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x10010040 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x10010060 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x10010080 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x100100a0 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x100100c0 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x10010000 (.data)
Hexadecimal Addresses
Hexadecimal Values
ASCII
Messages
Run I/O
Clear
-- program is finished running (0) --
Registers
Floating Point
Control and Status
Name
Number
Value
zero 0 0x00000000
ra 1 0x00000000
sp 2 0x7fffeffc
gp 3 0x10008000
tp 4 0x00000000
t0 5 0x00000000
t1 6 0x00000000
t2 7 0x00000000
s0 8 0x00000000
s1 9 0x80000000
a0 10 0x10010000
a1 11 0x00000005
a2 12 0x00000000
a3 13 0x00000000
a4 14 0x00000000
a5 15 0x00000000
a6 16 0x00000000
a7 17 0x00000000
s2 18 0x00000000
s3 19 0x00000000
s4 20 0x00000000
s5 21 0x00000000
s6 22 0x00000000
s7 23 0x00000000
s8 24 0x00000000
s9 25 0x00000000
s10 26 0x00000000
s11 27 0x00000000
t3 28 0x10010000
t4 29 0x00000001

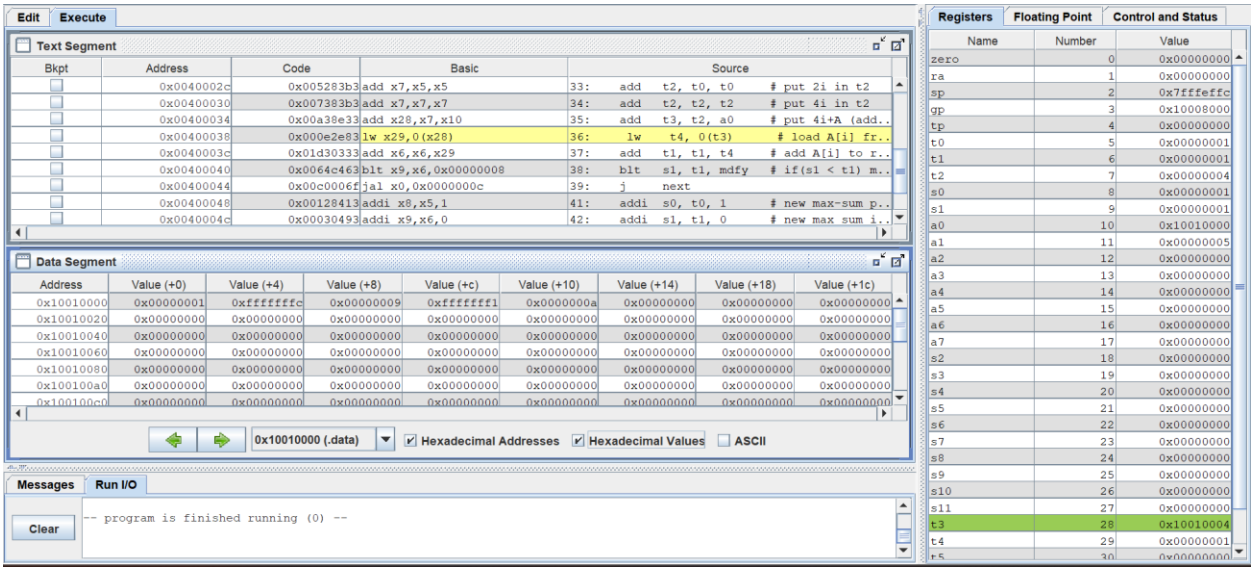


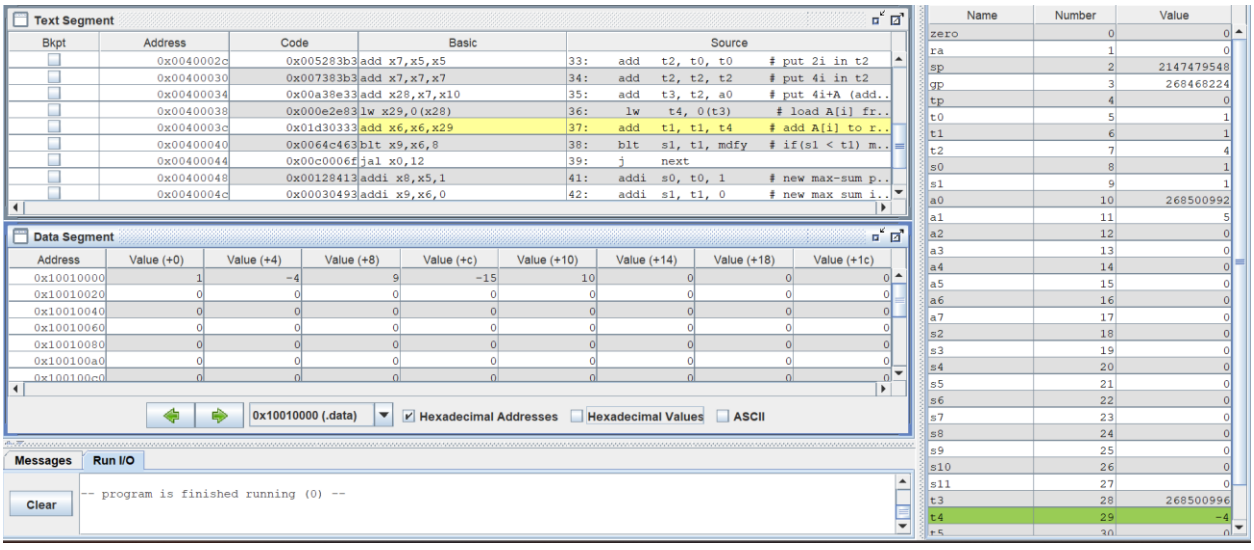


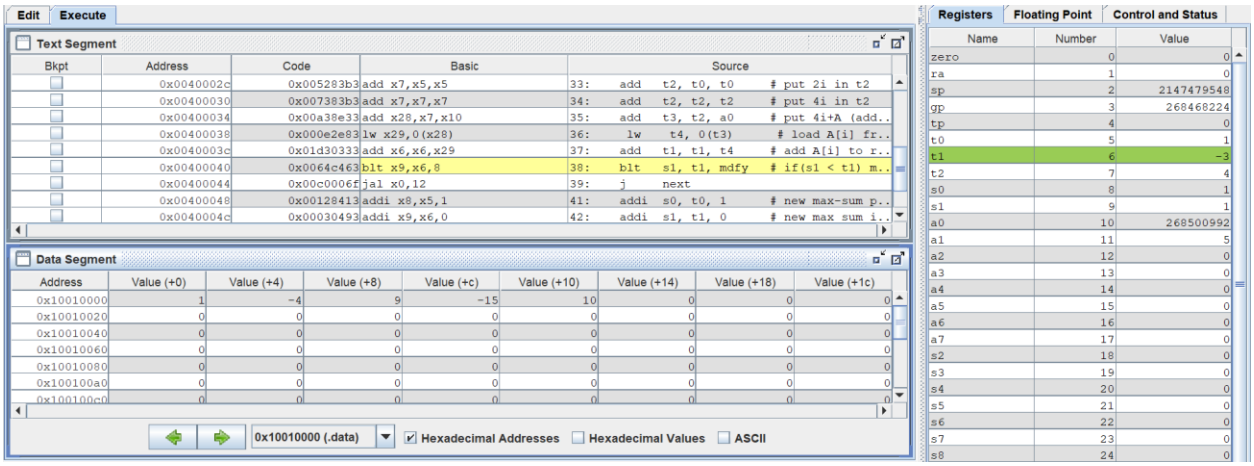
In the second iteration, i=t0=1, t2=4 and the register t3 gets the address of A[1], which is 0x10010004. Then, t4=A[1]= -4. The running sum value in t1 is equal to -3 (A[0]+A[1]). Because s1>t1, s1 remains.



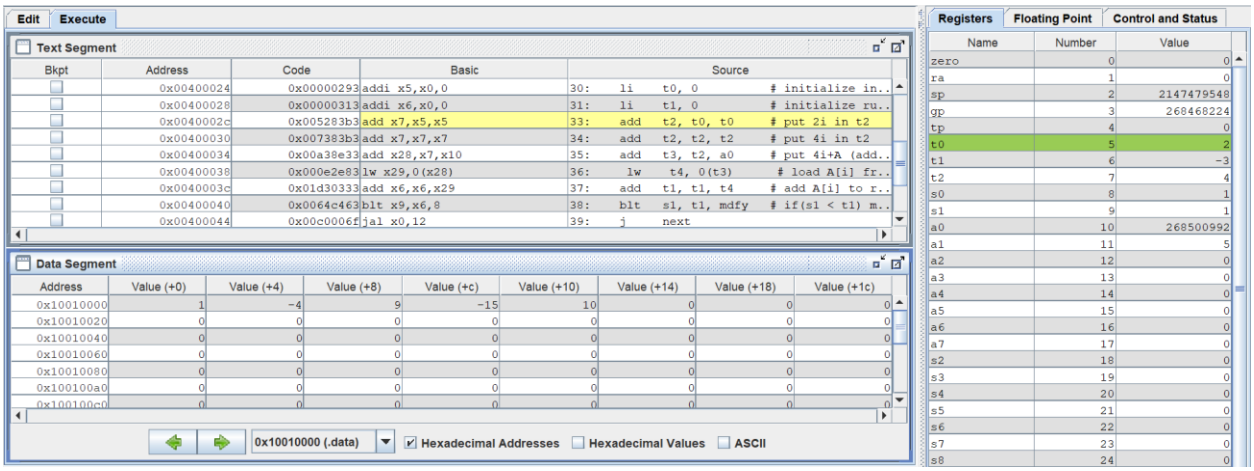

Edit
Execute
Text Segment
Bkpt Address Code Basic Source
0x0040002c 0x005283b3 add x7,x5,x5 33: add t2, t0, t0 # put 2i in t2
0x00400030 0x007383b3 add x7,x7,x7 34: add t2, t2, t2 # put 4i in t2
0x00400034 0x00a38e33 add x28,x7,x10 35: add t3, t2, a0 # put 4i+A (add..
0x00400038 0x000e2e83 lw x29,0(x28) 36: lw t4, 0(t3) # load A[i] fr..
0x0040003c 0x01d30333 add x6,x6,x29 37: add t1, t1, t4 # add A[i] to r..
0x00400040 0x0064c463 blt x9,x6,8 38: blt s1, t1, mdfy # if(s1 < t1) m..
0x00400044 0x00c0006f jal x0,12 39: j next
0x00400048 0x00128413 addi x8,x5,1 41: addi s0, t0, 1 # new max-sum p..
0x0040004c 0x00030493 addi x9,x6,0 42: addi s1, t1, 0 # new max sum i..
Data Segment
Address Value (+0) Value (+4) Value (+8) Value (+c) Value (+10) Value (+14) Value (+18) Value (+1c)
0x10010000 1 -4 9 -15 10 0 0 0
0x10010020 0 0 0 0 0 0 0 0
0x10010040 0 0 0 0 0 0 0 0
0x10010060 0 0 0 0 0 0 0 0
0x10010080 0 0 0 0 0 0 0 0
0x100100a0 0 0 0 0 0 0 0 0
0x100100c0 0 0 0 0 0 0 0 0
0x10010000 (.data)
Hexadecimal Addresses
Hexadecimal Values
ASCII
Registers
Floating Point
Control and Status
Name Number Value
zero 0 0
ra 1 0
sp 2 2147479548
gp 3 268468224
tp 4 0
t0 5 1
t1 6 1
t2 7 4
s0 8 1
s1 9 1
a0 10 268500992
a1 11 5
a2 12 0
a3 13 0
a4 14 0
a5 15 0
a6 16 0
a7 17 0
s2 18 0
s3 19 0
s4 20 0
s5 21 0
s6 22 0
s7 23 0
s8 24 0

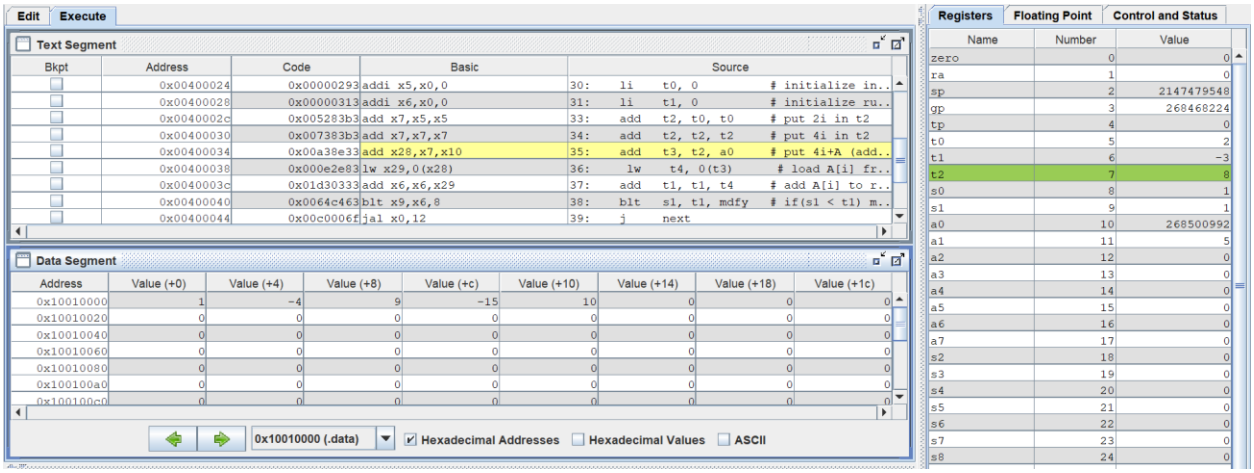

Edit
Execute
Text Segment
Bkpt Address Code Basic Source
0x0040002c 0x005283b3 add x7,x5,x5 33: add t2, t0, t0 # put 2i in t2
0x00400030 0x007383b3 add x7,x7,x7 34: add t2, t2, t2 # put 4i in t2
0x00400034 0x00a38e33 add x28,x7,x10 35: add t3, t2, a0 # put 4i+A (add..
0x00400038 0x000e2e83 lw x29,0(x28) 36: lw t4, 0(t3) # load A[i] fr..
0x0040003c 0x01d30333 add x6,x6,x29 37: add t1, t1, t4 # add A[i] to r..
0x00400040 0x0064c463 blt x9,x6,0x00000008 38: blt s1, t1, mdfy # if(s1 < t1) m..
0x00400044 0x00c0006f jal x0,0x0000000c 39: j next
0x00400048 0x00128413 addi x8,x5,1 41: addi s0, t0, 1 # new max-sum p..
0x0040004c 0x00030493 addi x9,x6,0 42: addi s1, t1, 0 # new max sum i..
Data Segment
Address Value (+0) Value (+4) Value (+8) Value (+c) Value (+10) Value (+14) Value (+18) Value (+1c)
0x10010000 0x00000001 0xfffffffc 0x00000009 0xfffffff1 0x0000000a 0x00000000 0x00000000 0x00000000
0x10010020 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x10010040 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x10010060 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x10010080 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x100100a0 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x100100c0 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x10010000 (.data)
Hexadecimal Addresses
Hexadecimal Values
ASCII
Messages
Run I/O
Clear
-- program is finished running (0) --
Registers
Floating Point
Control and Status
Name Number Value
zero 0 0x00000000
ra 1 0x00000000
sp 2 0x7fffeffc
gp 3 0x10008000
tp 4 0x00000000
t0 5 0x00000001
t1 6 0x00000001
t2 7 0x00000004
s0 8 0x00000001
s1 9 0x00000001
a0 10 0x10010000
a1 11 0x00000005
a2 12 0x00000000
a3 13 0x00000000
a4 14 0x00000000
a5 15 0x00000000
a6 16 0x00000000
a7 17 0x00000000
s2 18 0x00000000
s3 19 0x00000000
s4 20 0x00000000
s5 21 0x00000000
s6 22 0x00000000
s7 23 0x00000000
s8 24 0x00000000
s9 25 0x00000000
s10 26 0x00000000
s11 27 0x00000000
t3 28 0x10010004
t4 29 0x00000001

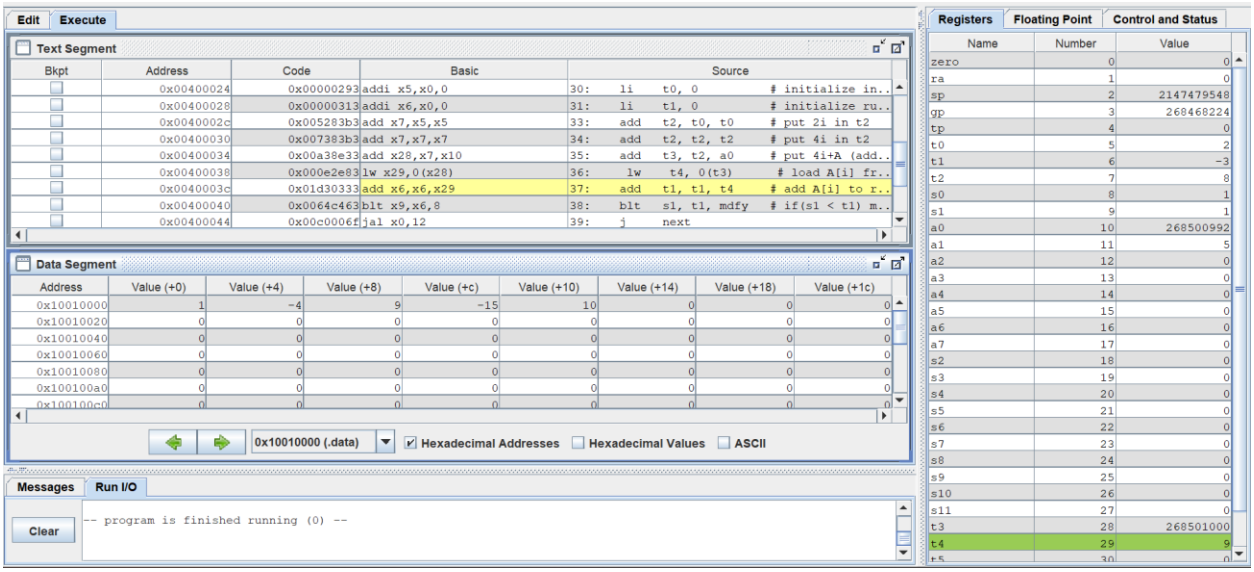

Text Segment
Bkpt Address Code Basic Source
0x0040002c 0x005283b3 add x7,x5,x5 33: add t2, t0, t0 # put 2i in t2
0x00400030 0x007383b3 add x7,x7,x7 34: add t2, t2, t2 # put 4i in t2
0x00400034 0x00a38e33 add x28,x7,x10 35: add t3, t2, a0 # put 4i+A (add..
0x00400038 0x000e2e83 lw x29,0(x28) 36: lw t4, 0(t3) # load A[i] fr..
0x0040003c 0x01d30333 add x6,x6,x29 37: add t1, t1, t4 # add A[i] to r..
0x00400040 0x0064c463 blt x9,x6,8 38: blt s1, t1, mdfy # if(s1 < t1) m..
0x00400044 0x00c0006f jal x0,12 39: j next
0x00400048 0x00128413 addi x8,x5,1 41: addi s0, t0, 1 # new max-sum p..
0x0040004c 0x00030493 addi x9,x6,0 42: addi s1, t1, 0 # new max sum i..
Data Segment
Address Value (+0) Value (+4) Value (+8) Value (+c) Value (+10) Value (+14) Value (+18) Value (+1c)
0x10010000 1 -4 9 -15 10 0 0 0
0x10010020 0 0 0 0 0 0 0 0
0x10010040 0 0 0 0 0 0 0 0
0x10010060 0 0 0 0 0 0 0 0
0x10010080 0 0 0 0 0 0 0 0
0x100100a0 0 0 0 0 0 0 0 0
0x100100c0 0 0 0 0 0 0 0 0
0x10010000 (.data)
Hexadecimal Addresses
Hexadecimal Values
ASCII
Messages
Run I/O
Clear
-- program is finished running (0) --
Name Number Value
zero 0 0
ra 1 0
sp 2 2147479548
gp 3 268468224
tp 4 0
t0 5 1
t1 6 1
t2 7 4
s0 8 1
s1 9 1
a0 10 268500992
a1 11 5
a2 12 0
a3 13 0
a4 14 0
a5 15 0
a6 16 0
a7 17 0
s2 18 0
s3 19 0
s4 20 0
s5 21 0
s6 22 0
s7 23 0
s8 24 0
s9 25 0
s10 26 0
s11 27 0
t3 28 268500996
t4 29 -4

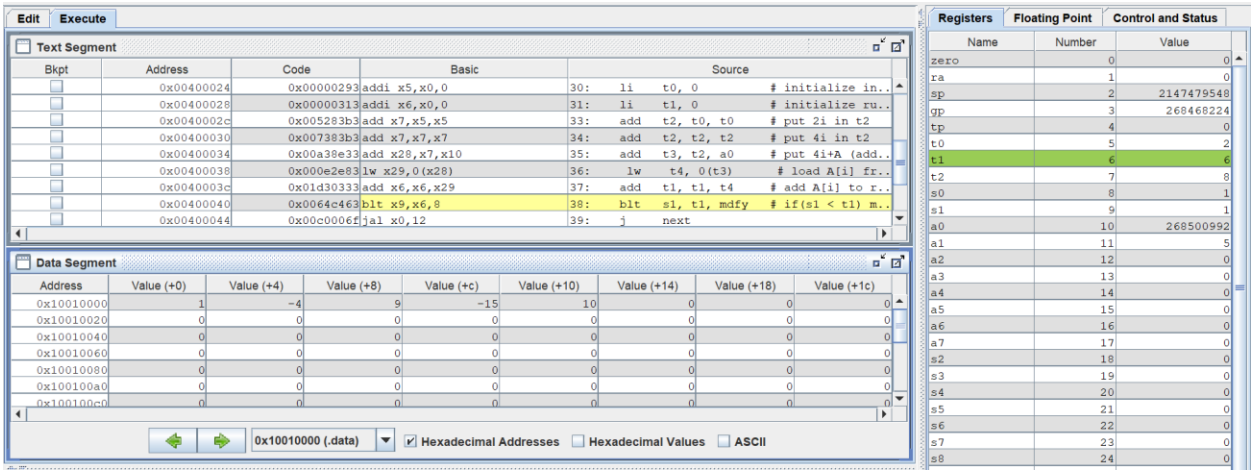


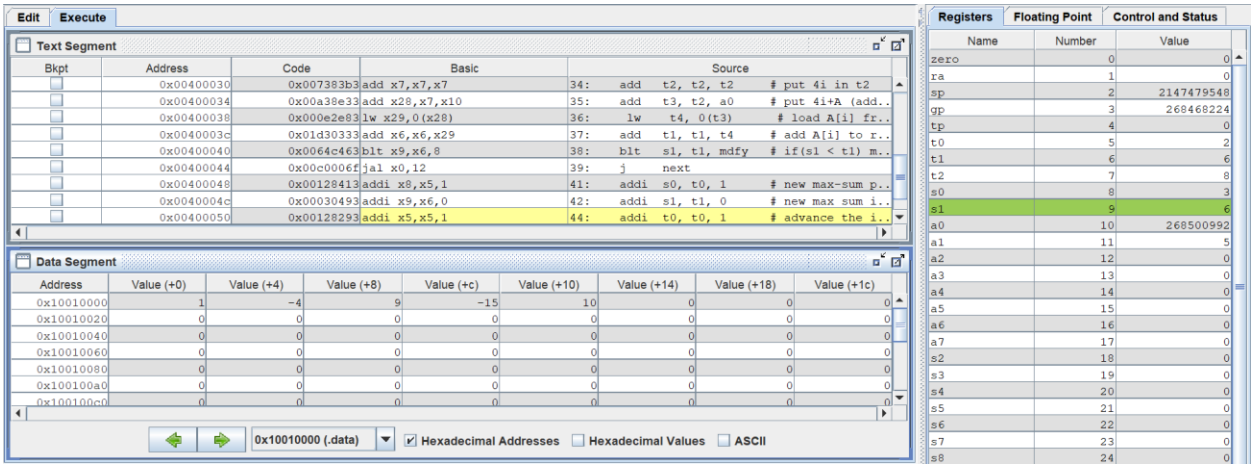
In the third iteration, i=t0=2, t2=8 and the register t3 gets the address of A[2], which is 0x10010008. Then, t4=A[2]= 9. The running sum value in t1 is equal to 6 (A[0]+A[1]+A[2]). Because s1=-3<t1=6, s1 gets new value of 6.



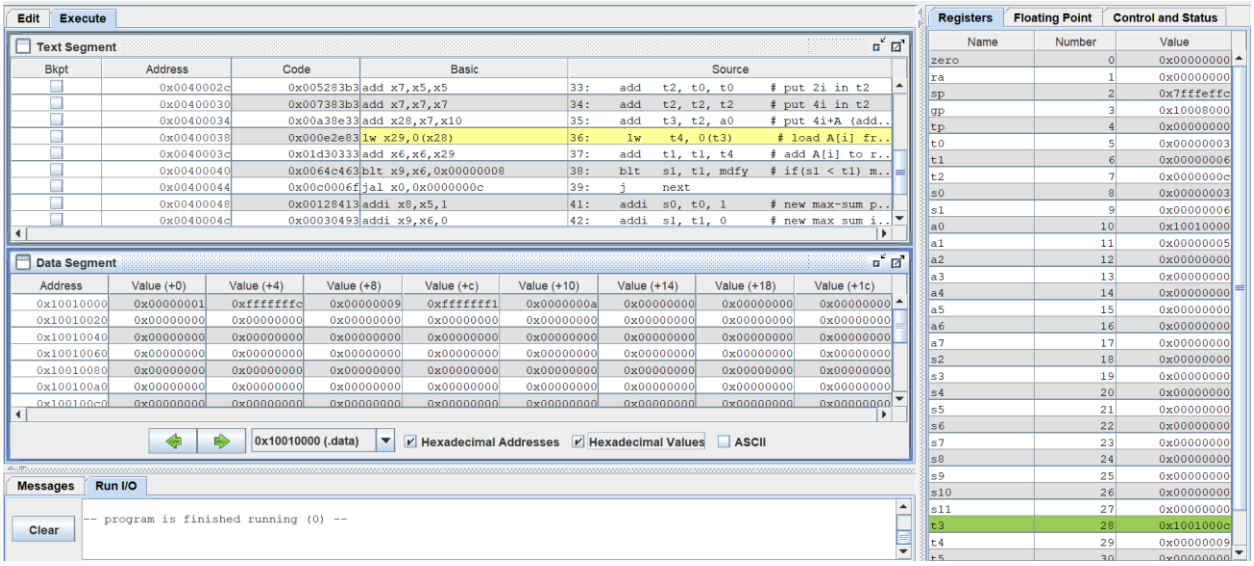


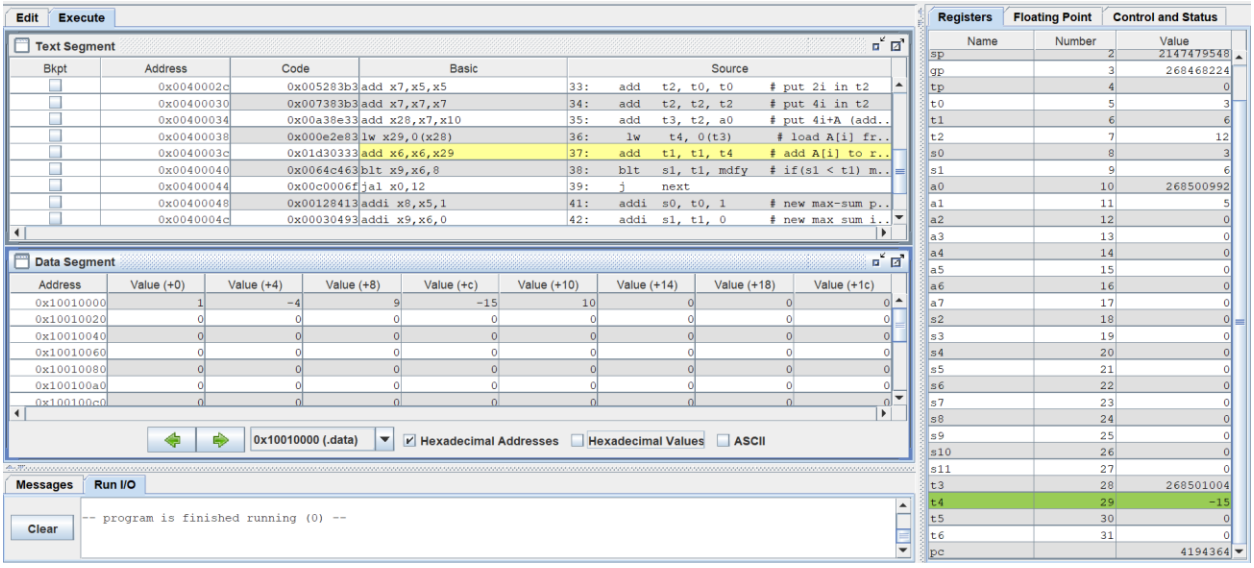


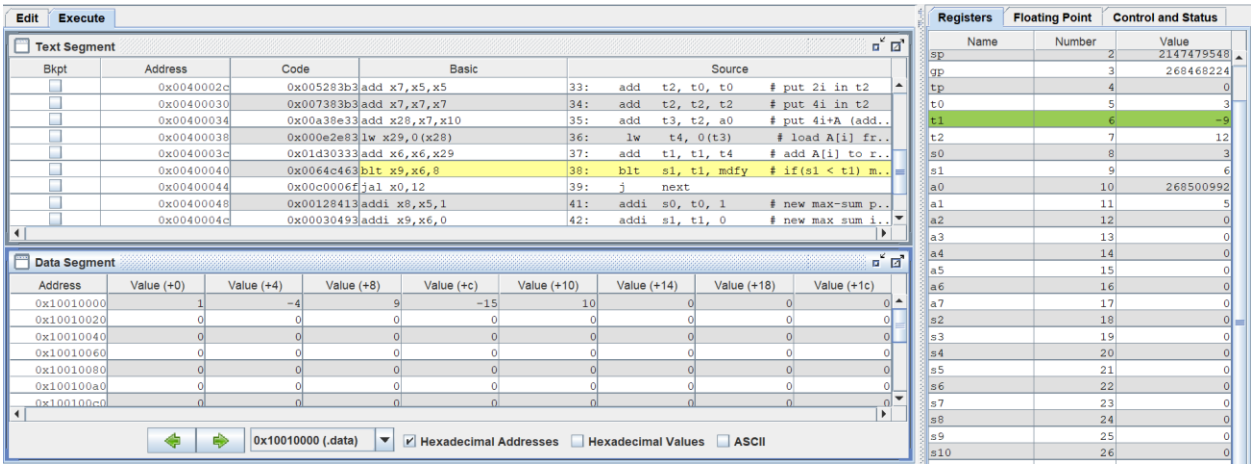




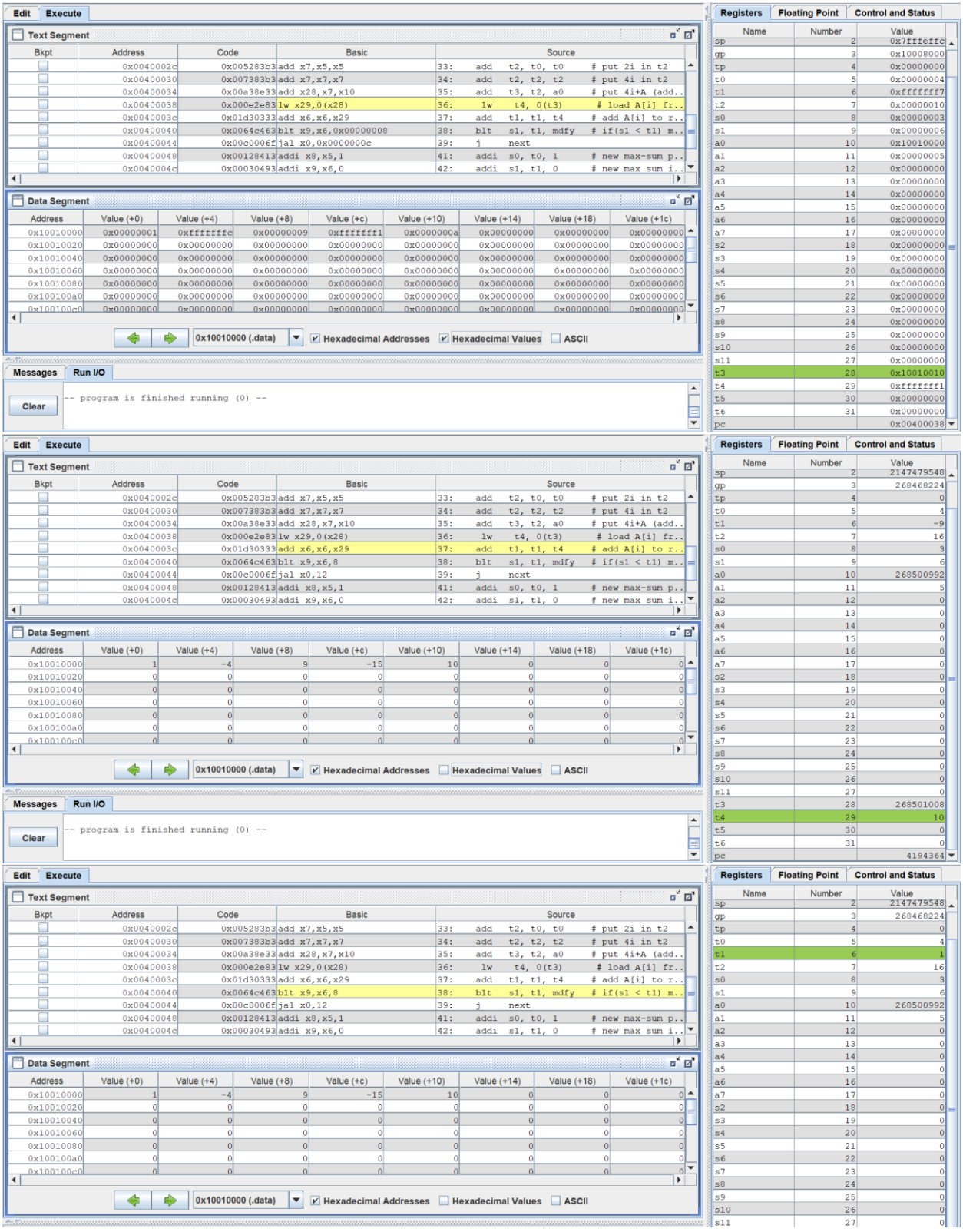
In the fourth iteration, i=t0=3, t2=12 and the register t3 gets the address of A[3], which is 0x1001000c. Then, t4=A[3]= -15. The running sum value in t1 is equal to -9 (A[0]+A[1]+A[2]+A[3]). Because s1>t1, s1 remains.


Edit
Execute
Text Segment
Bkpt
Address
Code
Basic
Source
0x0040002c 0x005283b3 add x7,x5,x5 33: add t2, t0, t0 # put 2i in t2
0x00400030 0x007383b3 add x7,x7,x7 34: add t2, t2, t2 # put 4i in t2
0x00400034 0x00a38e33 add x28,x7,x10 35: add t3, t2, a0 # put 4i+A (add..
0x00400038 0x000e2e83 lw x29,0(x28) 36: lw t4, 0(t3) # load A[i] fr..
0x0040003c 0x01d30333 add x6,x6,x29 37: add t1, t1, t4 # add A[i] to r..
0x00400040 0x0064c463 blt x9,x6,0x00000008 38: blt s1, t1, mdfy # if(s1 < t1) m..
0x00400044 0x00c0006f jal x0,0x0000000c 39: j next
0x00400048 0x00128413 addi x8,x5,1 41: addi s0, t0, 1 # new max-sum p..
0x0040004c 0x00030493 addi x9,x6,0 42: addi s1, t1, 0 # new max sum i..
Data Segment
Address
Value (+0)
Value (+4)
Value (+8)
Value (+c)
Value (+10)
Value (+14)
Value (+18)
Value (+1c)
0x10010000 0x00000001 0xfffffffc 0x00000009 0xfffffff1 0x0000000a 0x00000000 0x00000000 0x00000000
0x10010020 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x10010040 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x10010060 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x10010080 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x100100a0 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x100100c0 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000 0x00000000
0x10010000 (.data)
Hexadecimal Addresses
Hexadecimal Values
ASCII
Messages
Run I/O
Clear
-- program is finished running (0) --
Registers
Floating Point
Control and Status
Name
Number
Value
zero 0 0x00000000
ra 1 0x00000000
sp 2 0x7fffeffc
gp 3 0x10008000
tp 4 0x00000000
t0 5 0x00000003
t1 6 0x00000006
t2 7 0x0000000c
s0 8 0x00000003
s1 9 0x00000006
a0 10 0x10010000
a1 11 0x00000005
a2 12 0x00000000
a3 13 0x00000000
a4 14 0x00000000
a5 15 0x00000000
a6 16 0x00000000
a7 17 0x00000000
s2 18 0x00000000
s3 19 0x00000000
s4 20 0x00000000
s5 21 0x00000000
s6 22 0x00000000
s7 23 0x00000000
s8 24 0x00000000
s9 25 0x00000000
s10 26 0x00000000
s11 27 0x00000000
t3 28 0x1001000c
t4 29 0x00000009


Edit
Execute
Text Segment
Bkpt
Address
Code
Basic
Source
0x0040002c 0x005283b3 add x7,x5,x5 33: add t2, t0, t0 # put 2i in t2
0x00400030 0x007383b3 add x7,x7,x7 34: add t2, t2, t2 # put 4i in t2
0x00400034 0x00a38e33 add x28,x7,x10 35: add t3, t2, a0 # put 4i+A (add..
0x00400038 0x000e2e83 lw x29,0(x28) 36: lw t4, 0(t3) # load A[i] fr..
0x0040003c 0x01d30333 add x6,x6,x29 37: add t1, t1, t4 # add A[i] to r..
0x00400040 0x0064c463 blt x9,x6,8 38: blt s1, t1, mdfy # if(s1 < t1) m..
0x00400044 0x00c0006f jal x0,12 39: j next
0x00400048 0x00128413 addi x8,x5,1 41: addi s0, t0, 1 # new max-sum p..
0x0040004c 0x00030493 addi x9,x6,0 42: addi s1, t1, 0 # new max sum i..
Data Segment
Address
Value (+0)
Value (+4)
Value (+8)
Value (+c)
Value (+10)
Value (+14)
Value (+18)
Value (+1c)
0x10010000 1 -4 9 -15 10 0 0 0
0x10010020 0 0 0 0 0 0 0 0
0x10010040 0 0 0 0 0 0 0 0
0x10010060 0 0 0 0 0 0 0 0
0x10010080 0 0 0 0 0 0 0 0
0x100100a0 0 0 0 0 0 0 0 0
0x100100c0 0 0 0 0 0 0 0 0
0x10010000 (.data)
Hexadecimal Addresses
Hexadecimal Values
ASCII
Messages
Run I/O
Clear
-- program is finished running (0) --
Registers
Floating Point
Control and Status
Name
Number
Value
sp 2 2147479548
gp 3 268468224
tp 4 0
t0 5 3
t1 6 6
t2 7 12
s0 8 3
s1 9 6
a0 10 268500992
a1 11 5
a2 12 0
a3 13 0
a4 14 0
a5 15 0
a6 16 0
a7 17 0
s2 18 0
s3 19 0
s4 20 0
s5 21 0
s6 22 0
s7 23 0
s8 24 0
s9 25 0
s10 26 0
s11 27 0
t3 28 268501004
t4 29 -15
t5 30 0
t6 31 0
pc 4194364


Edit
Execute
Text Segment
Bkpt
Address
Code
Basic
Source
0x0040002c 0x005283b3 add x7,x5,x5 33: add t2, t0, t0 # put 2i in t2
0x00400030 0x007383b3 add x7,x7,x7 34: add t2, t2, t2 # put 4i in t2
0x00400034 0x00a38e33 add x28,x7,x10 35: add t3, t2, a0 # put 4i+A (add..
0x00400038 0x000e2e83 lw x29,0(x28) 36: lw t4, 0(t3) # load A[i] fr..
0x0040003c 0x01d30333 add x6,x6,x29 37: add t1, t1, t4 # add A[i] to r..
0x00400040 0x0064c463 blt x9,x6,8 38: blt s1, t1, mdfy # if(s1 < t1) m..
0x00400044 0x00c0006f jal x0,12 39: j next
0x00400048 0x00128413 addi x8,x5,1 41: addi s0, t0, 1 # new max-sum p..
0x0040004c 0x00030493 addi x9,x6,0 42: addi s1, t1, 0 # new max sum i..
Data Segment
Address
Value (+0)
Value (+4)
Value (+8)
Value (+c)
Value (+10)
Value (+14)
Value (+18)
Value (+1c)
0x10010000 1 -4 9 -15 10 0 0 0
0x10010020 0 0 0 0 0 0 0 0
0x10010040 0 0 0 0 0 0 0 0
0x10010060 0 0 0 0 0 0 0 0
0x10010080 0 0 0 0 0 0 0 0
0x100100a0 0 0 0 0 0 0 0 0
0x100100c0 0 0 0 0 0 0 0 0
0x10010000 (.data)
Hexadecimal Addresses
Hexadecimal Values
ASCII
Registers
Floating Point
Control and Status
Name
Number
Value
sp 2 2147479548
gp 3 268468224
tp 4 0
t0 5 3
t1 6 -9
t2 7 12
s0 8 3
s1 9 6
a0 10 268500992
a1 11 5
a2 12 0
a3 13 0
a4 14 0
a5 15 0
a6 16 0
a7 17 0
s2 18 0
s3 19 0
s4 20 0
s5 21 0
s6 22 0
s7 23 0
s8 24 0
s9 25 0
s10 26 0

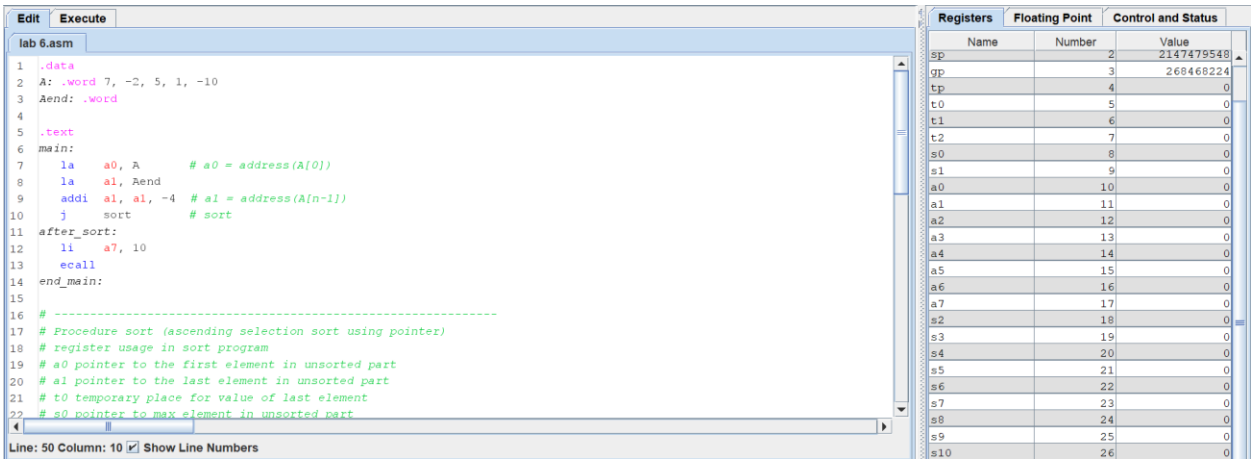
In the last iteration, i=t0=4, t2=16 and the register t3 gets the address of A[4], which is 0x10010010. Then, t4=A[4]= 10. The running sum value in t1 is equal to 1 (A[0]+A[1]+A[2]+A[3]+A[4]). Because s1>t1, s1 remains.



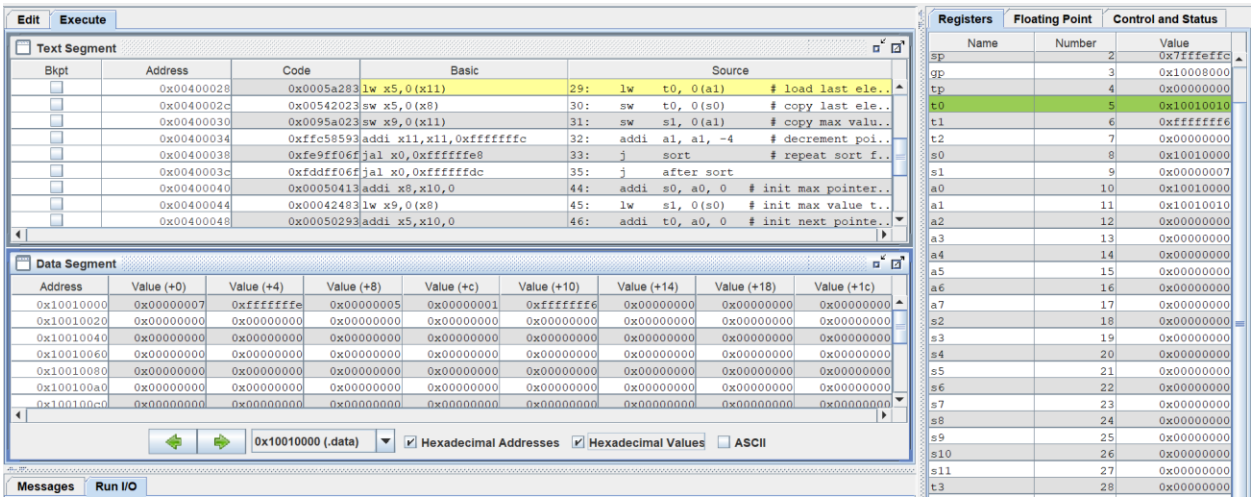
After running the program, the last rusult which is the max prefix-sum is 6. The program executes correctly according to algorithm.

Assignment 2: Create a new project that implements the program in Home Assignment 2. Initialize a new set of values for the array and compile it. Run the program step by step and observe the changes in the registers to verify that the program works according to the algorithm. Write an additional subprogram to print the array after each sorting pass.

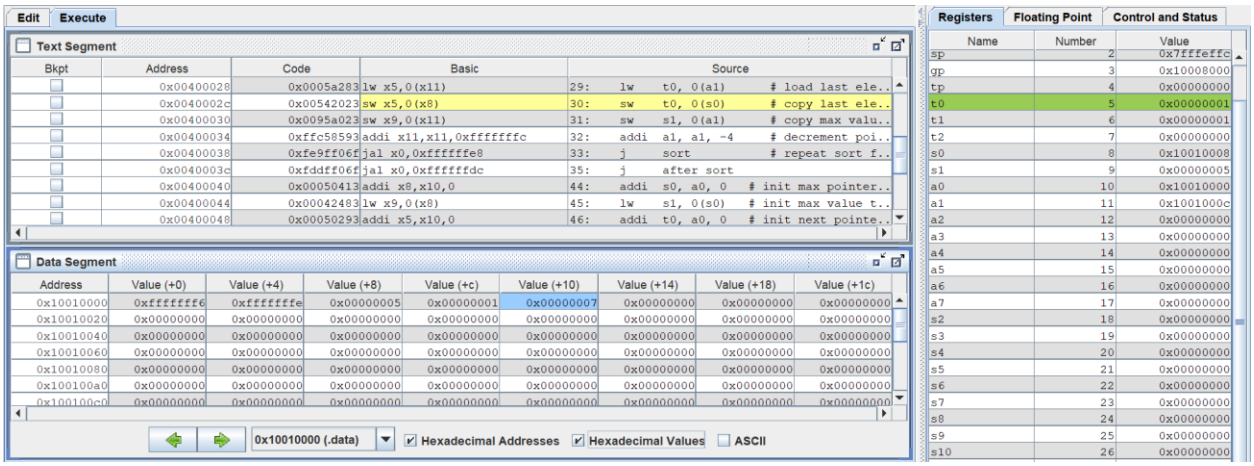
We initialize a new set of value {7, -2, 5, 1, -10} to label A.



In the first iteration of the first loop, the max element is 7 stored in register s1. The register s0 stores the address of max location which is 0x10010000. The two elements in s0 and a1 are swaped. The register a1 will point to A[3] which has address of 0x1001000c.



In the second iteration of the first loop, the max element is 5 stored in register s1. The register s0 stores the address of max location which is 0x10010008. The two elements in s0 and a1 are swaped. The register a1 will point to A[2] which has address of 0x10010008.



Edit | Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400028	0x0005a283	lw x5,0(x11)	29: lw t0, 0(a1) # load last ele..
	0x0040002c	0x00542023	sw x5,0(x8)	30: sw t0, 0(s0) # copy last ele..
	0x00400030	0x0095a023	sw x9,0(x11)	31: sw s1, 0(a1) # copy max valu..
	0x00400034	0xffc58593	addi x11,x11,0xfffffffc	32: addi a1, a1, -4 # decrement poi..
	0x00400038	0xfe9ff06f	jal x0,0xffffffe8	33: j sort # repeat sort f..
	0x0040003c	0xfdddff06f	jal x0,0xffffffdc	35: j after sort
	0x00400040	0x00050413	addi x8,x10,0	44: addi s0, a0, 0 # init max pointer..
	0x00400044	0x00042483	lw x9,0(x8)	45: lw s1, 0(s0) # init max value t..
	0x00400048	0x00050293	addi x5,x10,0	46: addi t0, a0, 0 # init next pointe..

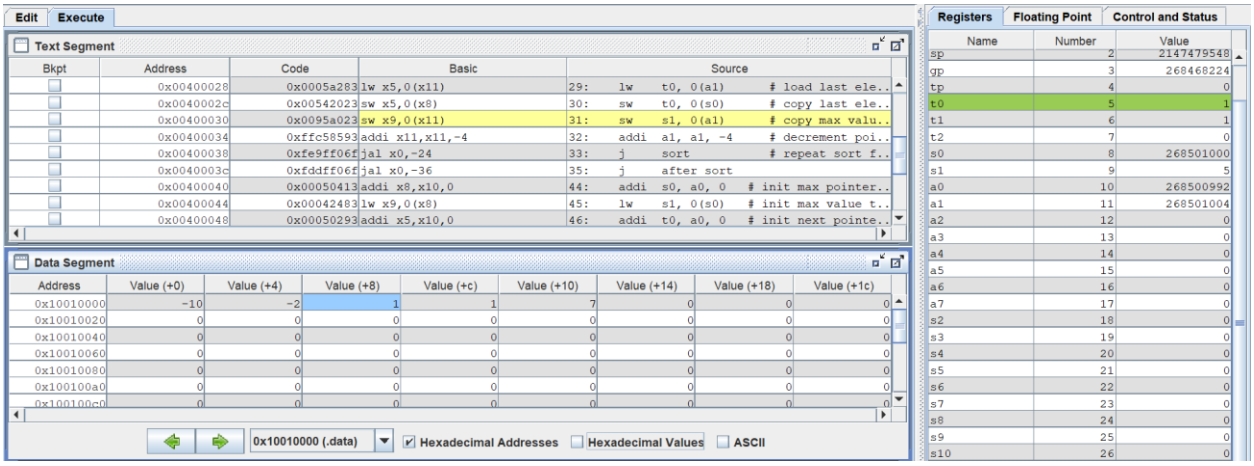
Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0xfffffff6	0xfffffffe	0x00000005	0x00000001	0x00000007	0x00000000	0x00000000	0x00000000
0x10010020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

0x10010000 (.data) ☑ Hexadecimal Addresses ☑ Hexadecimal Values ☐ ASCII

Registers | Floating Point | Control and Status

Name	Number	Value
sp	2	0x7fffeffc
gp	3	0x10008000
tp	4	0x00000000
t0	5	0x00000001
t1	6	0x00000001
t2	7	0x00000000
s0	8	0x10010008
s1	9	0x00000005
a0	10	0x10010000
a1	11	0x1001000c
a2	12	0x00000000
a3	13	0x00000000
a4	14	0x00000000
a5	15	0x00000000
a6	16	0x00000000
a7	17	0x00000000
s2	18	0x00000000
s3	19	0x00000000
s4	20	0x00000000
s5	21	0x00000000
s6	22	0x00000000
s7	23	0x00000000
s8	24	0x00000000
s9	25	0x00000000
s10	26	0x00000000



Edit | Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400028	0x0005a283	lw x5,0(x11)	29: lw t0, 0(a1) # load last ele..
	0x0040002c	0x00542023	sw x5,0(x8)	30: sw t0, 0(s0) # copy last ele..
	0x00400030	0x0095a023	sw x9,0(x11)	31: sw s1, 0(a1) # copy max valu..
	0x00400034	0xffc58593	addi x11,x11,-4	32: addi a1, a1, -4 # decrement poi..
	0x00400038	0xfe9ff06f	jal x0,-24	33: j sort # repeat sort f..
	0x0040003c	0xfdddff06f	jal x0,-36	35: j after sort
	0x00400040	0x00050413	addi x8,x10,0	44: addi s0, a0, 0 # init max pointer..
	0x00400044	0x00042483	lw x9,0(x8)	45: lw s1, 0(s0) # init max value t..
	0x00400048	0x00050293	addi x5,x10,0	46: addi t0, a0, 0 # init next pointe..

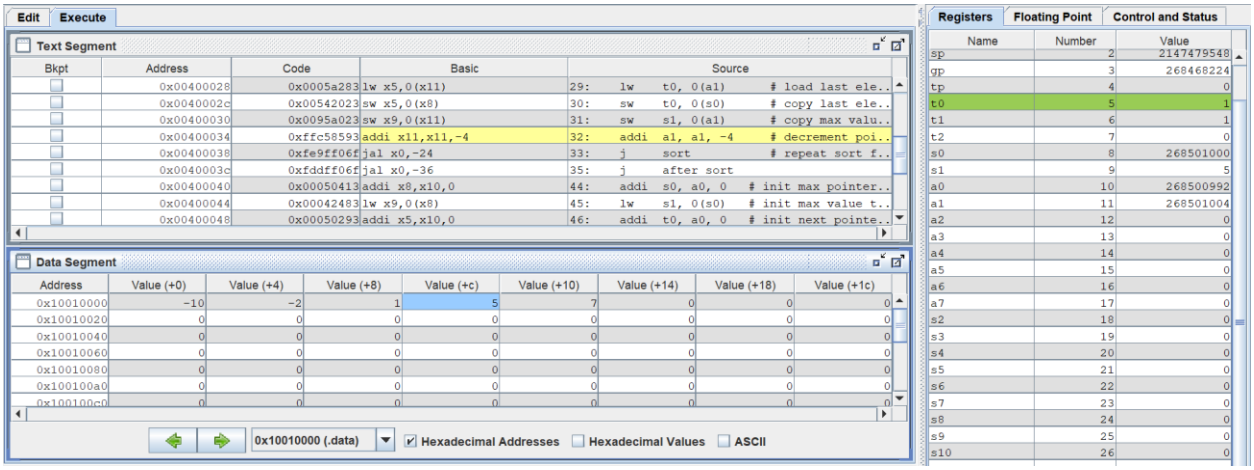
Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	-10	-2	1	1	7	0	0	0
0x10010020	0	0	0	0	0	0	0	0
0x10010040	0	0	0	0	0	0	0	0
0x10010060	0	0	0	0	0	0	0	0
0x10010080	0	0	0	0	0	0	0	0
0x100100a0	0	0	0	0	0	0	0	0
0x100100c0	0	0	0	0	0	0	0	0

0x10010000 (.data) ☑ Hexadecimal Addresses ☐ Hexadecimal Values ☐ ASCII

Registers | Floating Point | Control and Status

Name	Number	Value
sp	2	2147479548
gp	3	268468224
tp	4	0
t0	5	1
t1	6	1
t2	7	0
s0	8	268501000
s1	9	5
a0	10	268500992
a1	11	268501004
a2	12	0
a3	13	0
a4	14	0
a5	15	0
a6	16	0
a7	17	0
s2	18	0
s3	19	0
s4	20	0
s5	21	0
s6	22	0
s7	23	0
s8	24	0
s9	25	0
s10	26	0



Edit | Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400028	0x0005a283	lw x5,0(x11)	29: lw t0, 0(a1) # load last ele..
	0x0040002c	0x00542023	sw x5,0(x8)	30: sw t0, 0(s0) # copy last ele..
	0x00400030	0x0095a023	sw x9,0(x11)	31: sw s1, 0(a1) # copy max valu..
	0x00400034	0xffc58593	addi x11,x11,-4	32: addi a1, a1, -4 # decrement poi..
	0x00400038	0xfe9ff06f	jal x0,-24	33: j sort # repeat sort f..
	0x0040003c	0xfdddff06f	jal x0,-36	35: j after sort
	0x00400040	0x00050413	addi x8,x10,0	44: addi s0, a0, 0 # init max pointer..
	0x00400044	0x00042483	lw x9,0(x8)	45: lw s1, 0(s0) # init max value t..
	0x00400048	0x00050293	addi x5,x10,0	46: addi t0, a0, 0 # init next pointe..

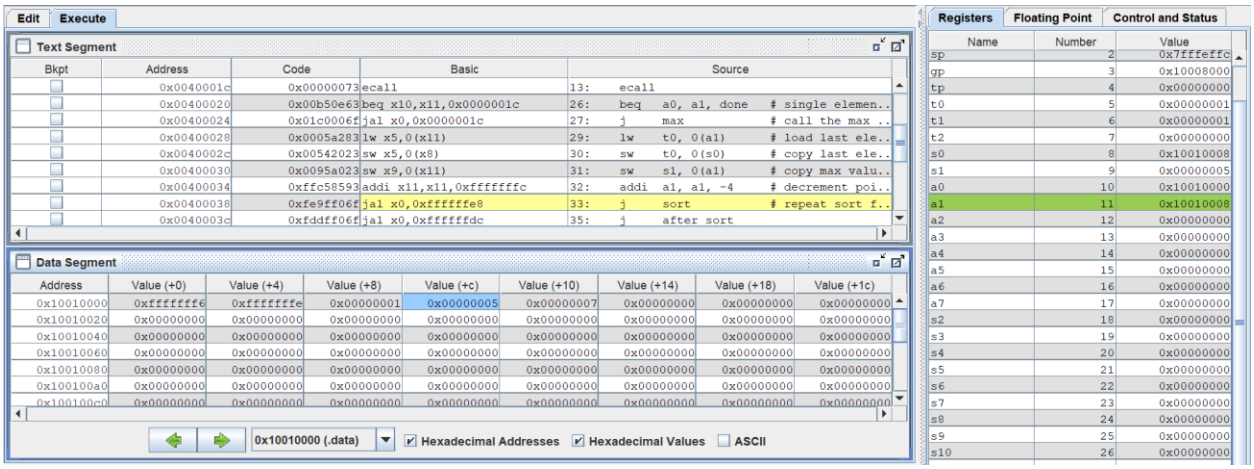
Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	-10	-2	1	5	7	0	0	0
0x10010020	0	0	0	0	0	0	0	0
0x10010040	0	0	0	0	0	0	0	0
0x10010060	0	0	0	0	0	0	0	0
0x10010080	0	0	0	0	0	0	0	0
0x100100a0	0	0	0	0	0	0	0	0
0x100100c0	0	0	0	0	0	0	0	0

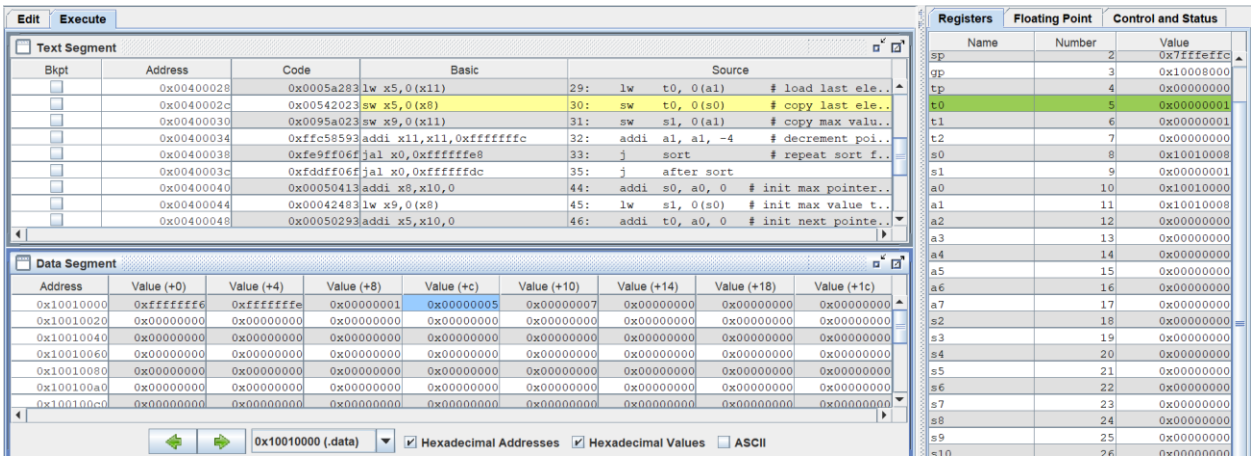
0x10010000 (.data) ☑ Hexadecimal Addresses ☐ Hexadecimal Values ☐ ASCII

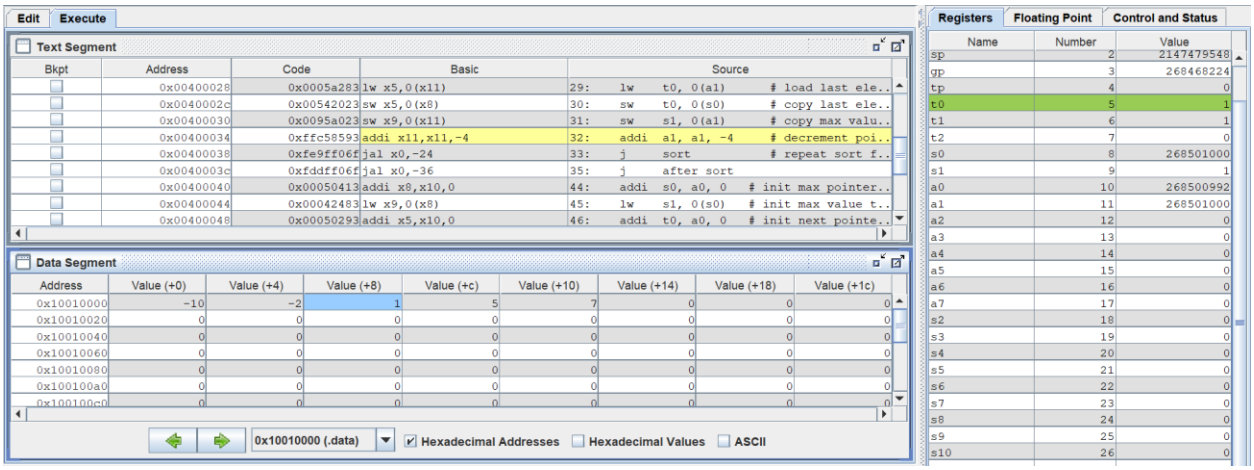
Registers | Floating Point | Control and Status

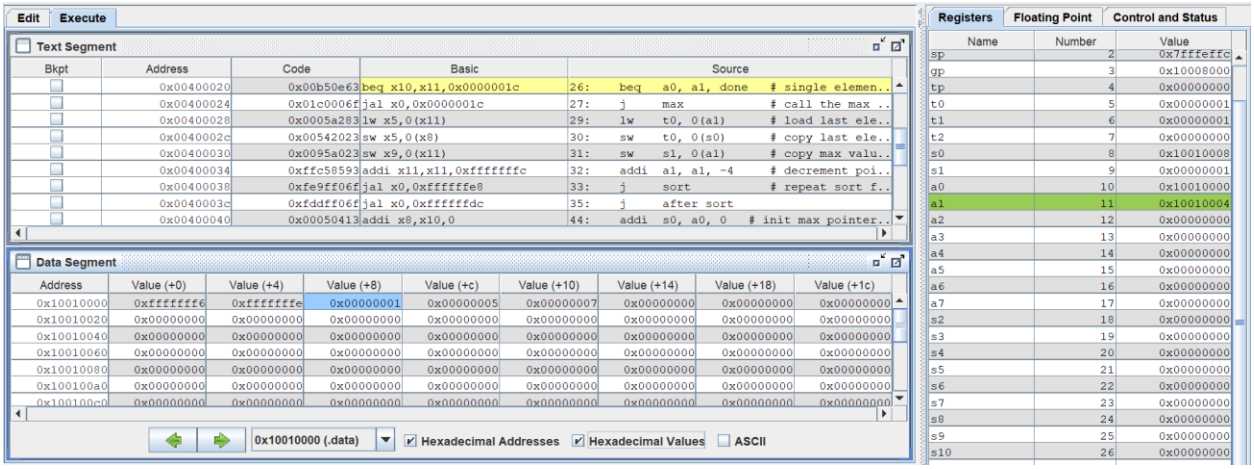
Name	Number	Value
sp	2	2147479548
gp	3	268468224
tp	4	0
t0	5	1
t1	6	1
t2	7	0
s0	8	268501000
s1	9	5
a0	10	268500992
a1	11	268501004
a2	12	0
a3	13	0
a4	14	0
a5	15	0
a6	16	0
a7	17	0
s2	18	0
s3	19	0
s4	20	0
s5	21	0
s6	22	0
s7	23	0
s8	24	0
s9	25	0
s10	26	0



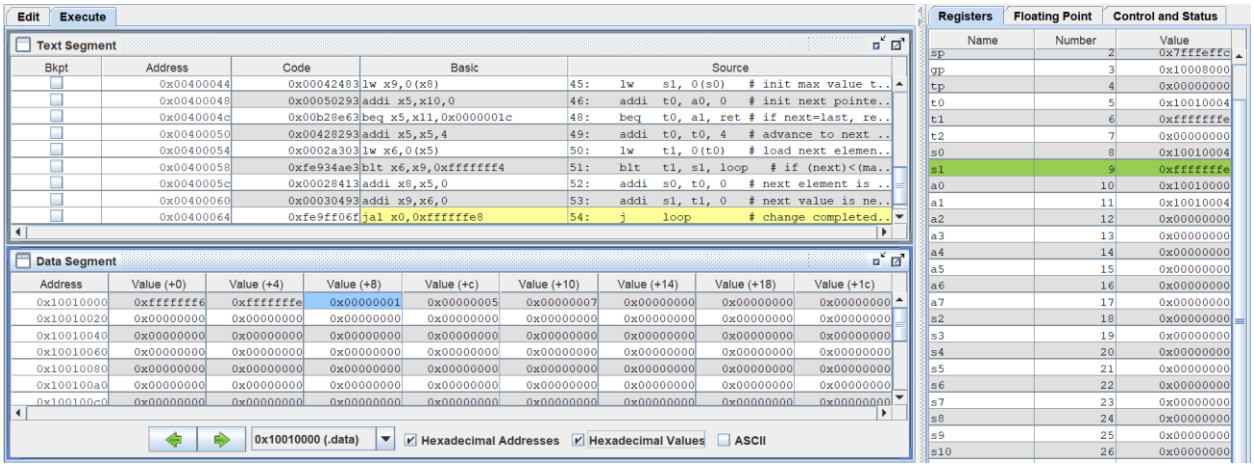
In the third iteration of the first loop, the max element is 1 stored in register s1. The register s0 stores the address of max location which is 0x10010008. The two elements in s0 and a1 are swaped. The register a1 will point to A[1] which has address of 0x10010004.

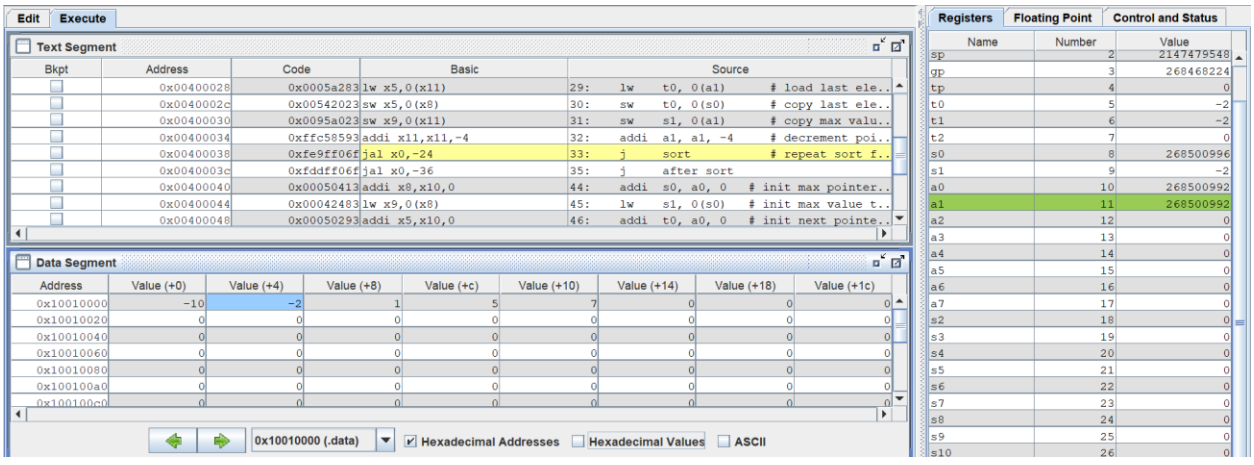


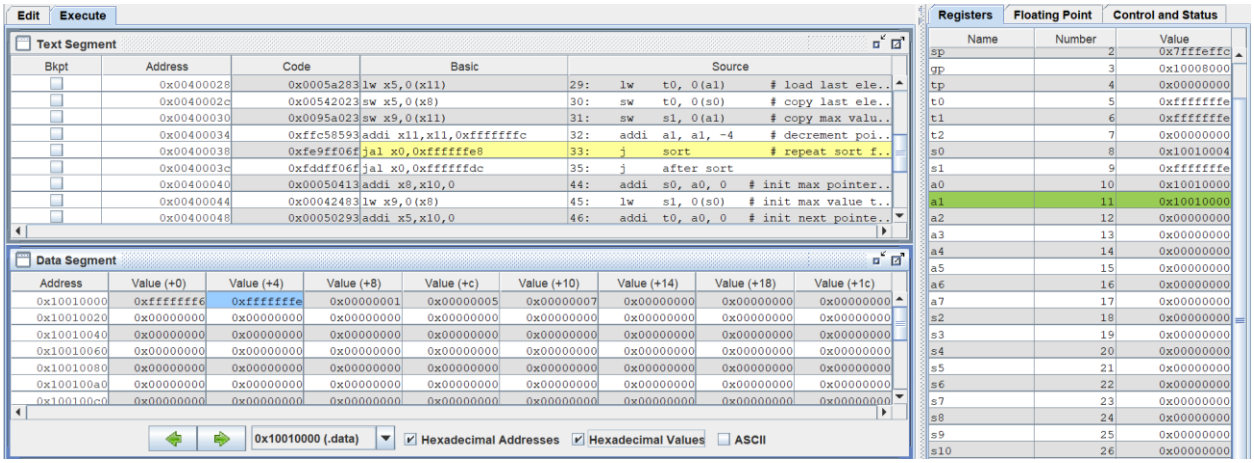




In the fourth iteration of the first loop, the max element is -2 stored in register s1. The register s0 stores the address of max location which is 0x10010004. The two elements in s0 and a1 are swaped. The register a1 will point to A[0] which has address of 0x10010000.

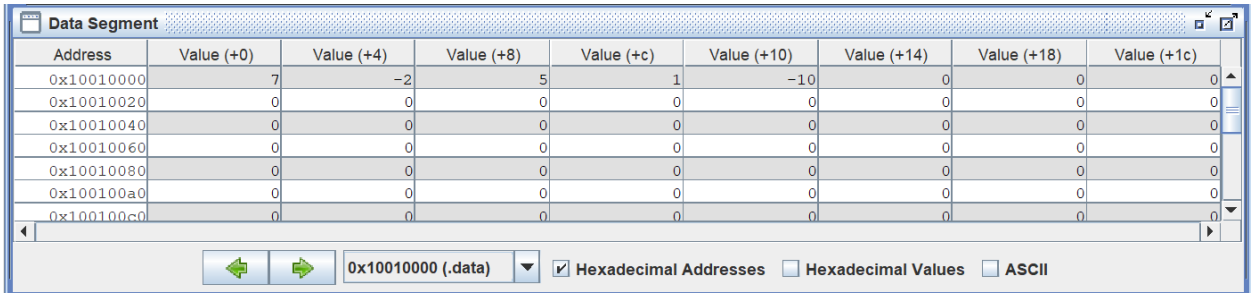




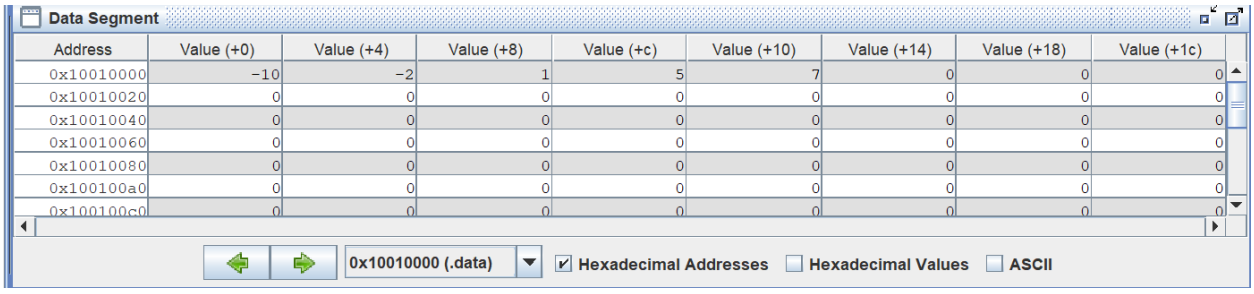


In the last iteration of the first loop, because a0=a1, the program finishes sorting. And we have an correctly ascending array according to algorithm.

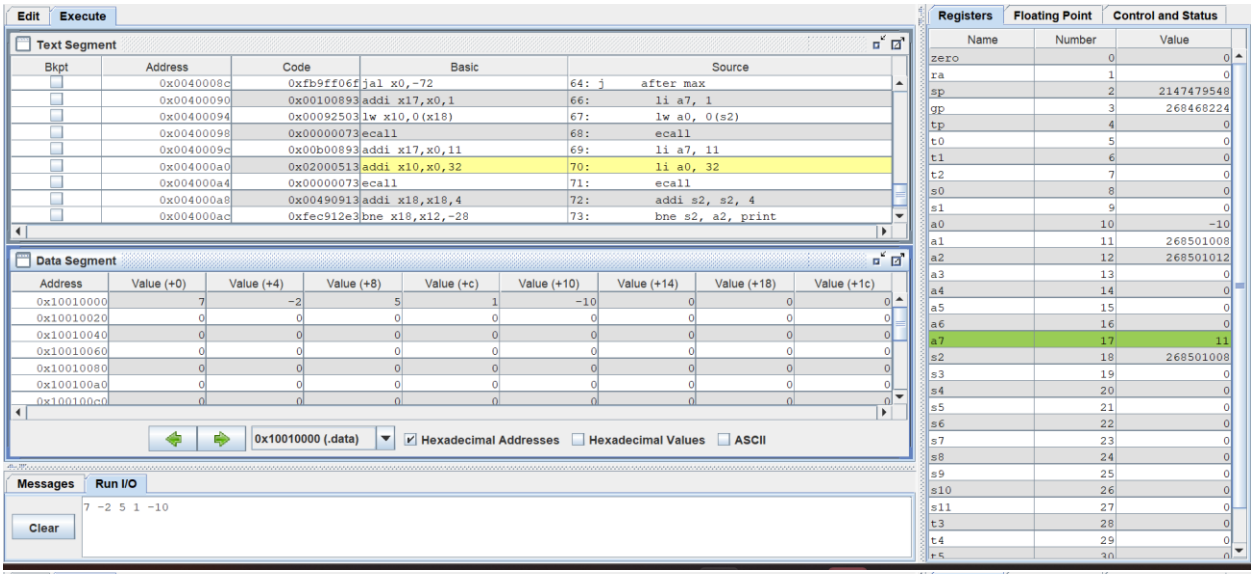
Before sorting:

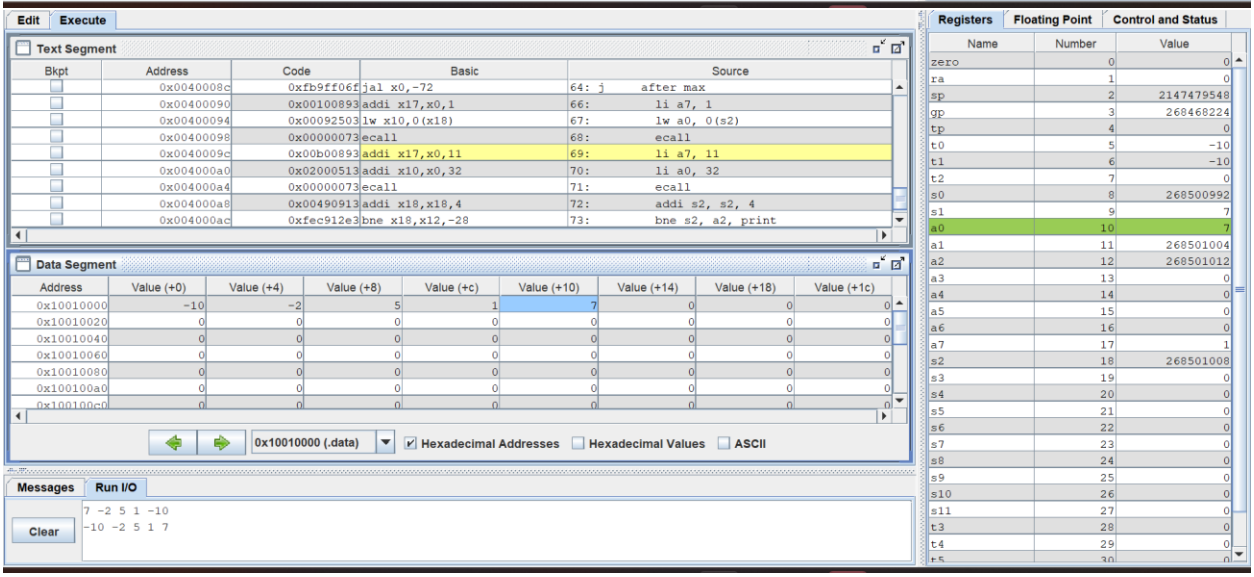


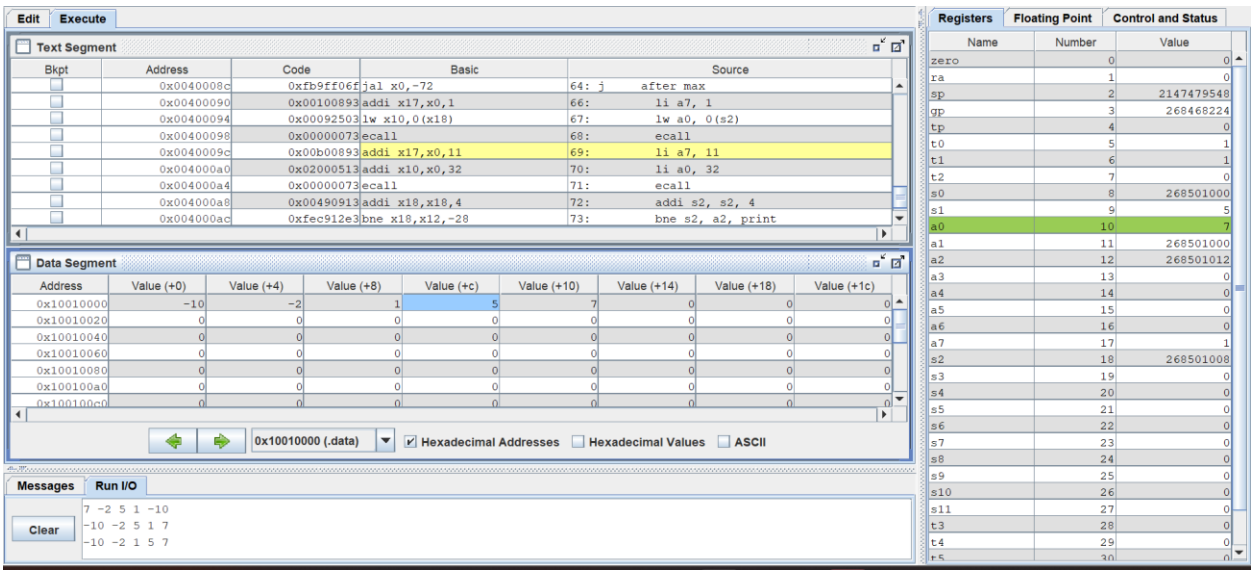
After sorting:

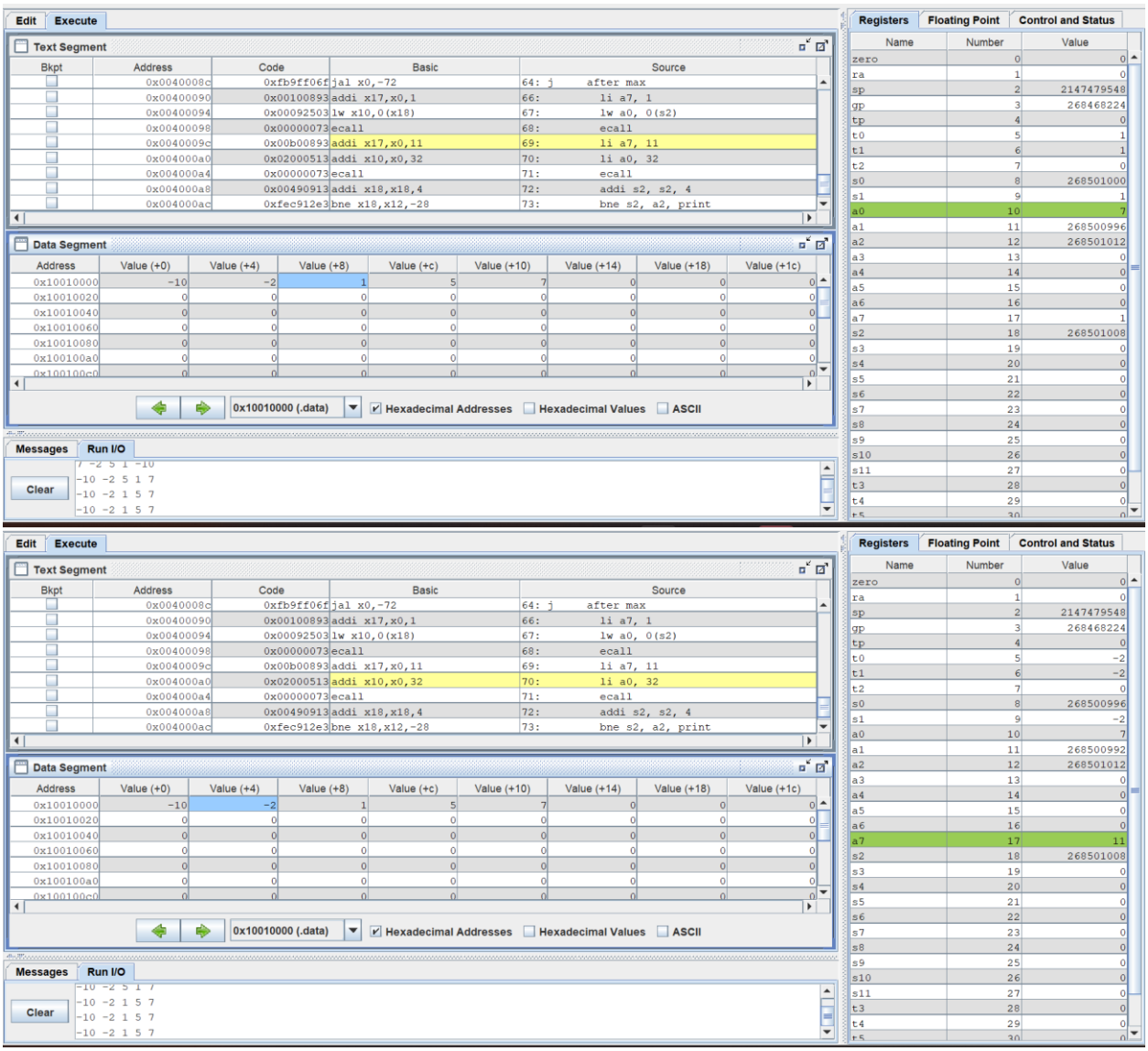


Result after adding subprogram to print array after each sorting pass:


Edit
Execute
Text Segment
Bkpt
Address
Code
Basic
Source
0x0040008c 0xfb9ff06f jal x0,-72 64: j after max
0x00400090 0x00100893 addi x17,x0,1 66: li a7, 1
0x00400094 0x00092503 lw x10,0(x18) 67: lw a0, 0(s2)
0x00400098 0x00000073 ecall 68: ecall
0x0040009c 0x00b00893 addi x17,x0,11 69: li a7, 11
0x004000a0 0x02000513 addi x10,x0,32 70: li a0, 32
0x004000a4 0x00000073 ecall 71: ecall
0x004000a8 0x00490913 addi x18,x18,4 72: addi s2, s2, 4
0x004000ac 0xfec912e3 bne x18,x12,-28 73: bne s2, a2, print
Data Segment
Address
Value (+0)
Value (+4)
Value (+8)
Value (+c)
Value (+10)
Value (+14)
Value (+18)
Value (+1c)
0x10010000 7 -2 5 1 -10 0 0 0
0x10010020 0 0 0 0 0 0 0 0
0x10010040 0 0 0 0 0 0 0 0
0x10010060 0 0 0 0 0 0 0 0
0x10010080 0 0 0 0 0 0 0 0
0x100100a0 0 0 0 0 0 0 0 0
0x100100c0 0 0 0 0 0 0 0 0
0x10010000 (.data)
Hexadecimal Addresses
Hexadecimal Values
ASCII
Messages
Run I/O
Clear
7 -2 5 1 -10
Registers
Floating Point
Control and Status
Name
Number
Value
zero 0 0
ra 1 0
sp 2 2147479548
gp 3 268468224
tp 4 0
t0 5 0
t1 6 0
t2 7 0
s0 8 0
s1 9 0
a0 10 -10
a1 11 268501008
a2 12 268501012
a3 13 0
a4 14 0
a5 15 0
a6 16 0
a7 17 11
s2 18 268501008
s3 19 0
s4 20 0
s5 21 0
s6 22 0
s7 23 0
s8 24 0
s9 25 0
s10 26 0
s11 27 0
t3 28 0
t4 29 0


Edit
Execute
Text Segment
Bkpt
Address
Code
Basic
Source
0x0040008c 0xfb9ff06f jal x0,-72 64: j after max
0x00400090 0x00100893 addi x17,x0,1 66: li a7, 1
0x00400094 0x00092503 lw x10,0(x18) 67: lw a0, 0(s2)
0x00400098 0x00000073 ecall 68: ecall
0x0040009c 0x00b00893 addi x17,x0,11 69: li a7, 11
0x004000a0 0x02000513 addi x10,x0,32 70: li a0, 32
0x004000a4 0x00000073 ecall 71: ecall
0x004000a8 0x00490913 addi x18,x18,4 72: addi s2, s2, 4
0x004000ac 0xfec912e3 bne x18,x12,-28 73: bne s2, a2, print
Data Segment
Address
Value (+0)
Value (+4)
Value (+8)
Value (+c)
Value (+10)
Value (+14)
Value (+18)
Value (+1c)
0x10010000 -10 -2 5 1 7 0 0 0
0x10010020 0 0 0 0 0 0 0 0
0x10010040 0 0 0 0 0 0 0 0
0x10010060 0 0 0 0 0 0 0 0
0x10010080 0 0 0 0 0 0 0 0
0x100100a0 0 0 0 0 0 0 0 0
0x100100c0 0 0 0 0 0 0 0 0
0x10010000 (.data)
Hexadecimal Addresses
Hexadecimal Values
ASCII
Messages
Run I/O
Clear
7 -2 5 1 -10
-10 -2 5 1 7
Registers
Floating Point
Control and Status
Name
Number
Value
zero 0 0
ra 1 0
sp 2 2147479548
gp 3 268468224
tp 4 0
t0 5 -10
t1 6 -10
t2 7 0
s0 8 268500992
s1 9 7
a0 10 7
a1 11 268501004
a2 12 268501012
a3 13 0
a4 14 0
a5 15 0
a6 16 0
a7 17 1
s2 18 268501008
s3 19 0
s4 20 0
s5 21 0
s6 22 0
s7 23 0
s8 24 0
s9 25 0
s10 26 0
s11 27 0
t3 28 0
t4 29 0


Edit
Execute
Text Segment
Bkpt
Address
Code
Basic
Source
0x0040008c 0xfb9ff06f jal x0,-72 64: j after max
0x00400090 0x00100893 addi x17,x0,1 66: li a7, 1
0x00400094 0x00092503 lw x10,0(x18) 67: lw a0, 0(s2)
0x00400098 0x00000073 ecall 68: ecall
0x0040009c 0x00b00893 addi x17,x0,11 69: li a7, 11
0x004000a0 0x02000513 addi x10,x0,32 70: li a0, 32
0x004000a4 0x00000073 ecall 71: ecall
0x004000a8 0x00490913 addi x18,x18,4 72: addi s2, s2, 4
0x004000ac 0xfec912e3 bne x18,x12,-28 73: bne s2, a2, print
Data Segment
Address
Value (+0)
Value (+4)
Value (+8)
Value (+c)
Value (+10)
Value (+14)
Value (+18)
Value (+1c)
0x10010000 -10 -2 1 5 7 0 0 0
0x10010020 0 0 0 0 0 0 0 0
0x10010040 0 0 0 0 0 0 0 0
0x10010060 0 0 0 0 0 0 0 0
0x10010080 0 0 0 0 0 0 0 0
0x100100a0 0 0 0 0 0 0 0 0
0x100100c0 0 0 0 0 0 0 0 0
0x10010000 (.data)
Hexadecimal Addresses
Hexadecimal Values
ASCII
Messages
Run I/O
Clear
7 -2 5 1 -10
-10 -2 5 1 7
-10 -2 1 5 7
Registers
Floating Point
Control and Status
Name
Number
Value
zero 0 0
ra 1 0
sp 2 2147479548
gp 3 268468224
tp 4 0
t0 5 1
t1 6 1
t2 7 0
s0 8 268501000
s1 9 5
a0 10 7
a1 11 268501000
a2 12 268501012
a3 13 0
a4 14 0
a5 15 0
a6 16 0
a7 17 1
s2 18 268501008
s3 19 0
s4 20 0
s5 21 0
s6 22 0
s7 23 0
s8 24 0
s9 25 0
s10 26 0
s11 27 0
t3 28 0
t4 29 0



Assignment 3: Write a program that implements the bubble sort algorithm.

The source code is as below:

Edit Execute

lab 6.asm

```
1  .data
2  A: .word 7, -2, 5, 1, -10, -65, 100, 23
3  Aend: .word
4
5  .text
6  main:
7     la    a0, A       # a0 = address(A[0])
8     la    a1, Aend
9     addi  a1, a1, -4  # a1 = address(A[n-1])
10    j     sort       # sort
11 after_sort:
12    li    a7, 10
13    ecall
14 end_main:
15
16 # ------------------------------------------------------------
17 # Procedure sort (ascending selection sort using pointer)
18 # register usage in sort program
19 # a0 pointer to the first element in unsorted part
20 # a1 pointer to the last element in unsorted part
21 # t0 temporary place for value of last element
22 # s0 pointer to max element in unsorted part
```

Line: 2 Column: 40 ☑ Show Line Numbers

Edit Execute

lab 6.asm

```
22 # s0 pointer to max element in unsorted part
23 # s1 value of max element in unsorted part
24 # ------------------------------------------------------------
25 sort:
26    la, s0, A
27    addi s0, s0, 4
28    beq   a0, a1, done   # single element list is sorted
29    j     loop            # call the max procedure
30 done:
31    j     after_sort
32
33 # ------------------------------------------------------------------
34 # Procedure max
35 # function: fax the value and address of max element in the list
36 # a0 pointer to first element
37 # a1 pointer to last element
38 # ------------------------------------------------------------------
39 loop:
40    lw    s1, 0(s0)     # get the value A[i+1]
41    addi s2, s0, -4     # get the value A[i]
42    lw s4, 0(s2)
43    bgt s4, s1, swap
```

Line: 2 Column: 40 ☑ Show Line Numbers

Edit Execute

lab 6.asm

```
37  # a1 pointer to last element
38  # ------------------------------------------------------------------------
39  loop:
40     lw    s1, 0(s0)      # get the value A[i+1]
41     addi s2, s0, -4      # get the value A[i]
42     lw s4, 0(s2)
43     bgt s4, s1, swap
44     j check
45  swap:
46          li s3, 1
47          addi t0, s1, 0
48          sw s4, 0(s0)
49          sw t0, 0(s2)
50  check:
51          addi s0, s0, 4
52          bgt s0, a1, loop1
53          j loop
54  loop1:
55          addi a1, a1, -4
56          j sort
57  end_sort:
```

Line: 2 Column: 40 Show Line Numbers

The result:

Before sorting:

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	7	-2	5	1	-10	-65	100	23
0x10010020	0	0	0	0	0	0	0	0
0x10010040	0	0	0	0	0	0	0	0
0x10010060	0	0	0	0	0	0	0	0
0x10010080	0	0	0	0	0	0	0	0
0x100100a0	0	0	0	0	0	0	0	0
0x100100c0	0	0	0	0	0	0	0	0

0x10010000 (.data) Hexadecimal Addresses Hexadecimal Values ASCII

After sorting:

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	-65	-10	-2	1	5	7	23	100
0x10010020	0	0	0	0	0	0	0	0
0x10010040	0	0	0	0	0	0	0	0
0x10010060	0	0	0	0	0	0	0	0
0x10010080	0	0	0	0	0	0	0	0
0x100100a0	0	0	0	0	0	0	0	0
0x100100c0	0	0	0	0	0	0	0	0

0x10010000 (.data) Hexadecimal Addresses Hexadecimal Values ASCII

Assignment 4: Write a program that implements the insertion sort algorithm.

The source code is as below:

Edit Execute

lab 6.asm*

```
1  .data
2  A: .word 7, -2, 5, 1, -10, -65, 100, 23
3  Aend: .word
4
5  .text
6  main:
7     la    s0, A       # s0 = address(A[0])
8     la a0, A
9     addi s0, s0, 4
10    la    a1, Aend
11    #addi  a1, a1, -4  # a1 = address(A[n-1])
12    j     sort       # sort
13 after_sort:
14    li    a7, 10
15    ecall
16 end_main:
17
18 sort:
19    beq   s0, a1, done   # if i>n
20    lw t1, 0(s0)
21    addi s1, s0, -4 # s1=i-1
22    j     loop
```

Line: 28 Column: 20 ☑ Show Line Numbers

Edit Execute

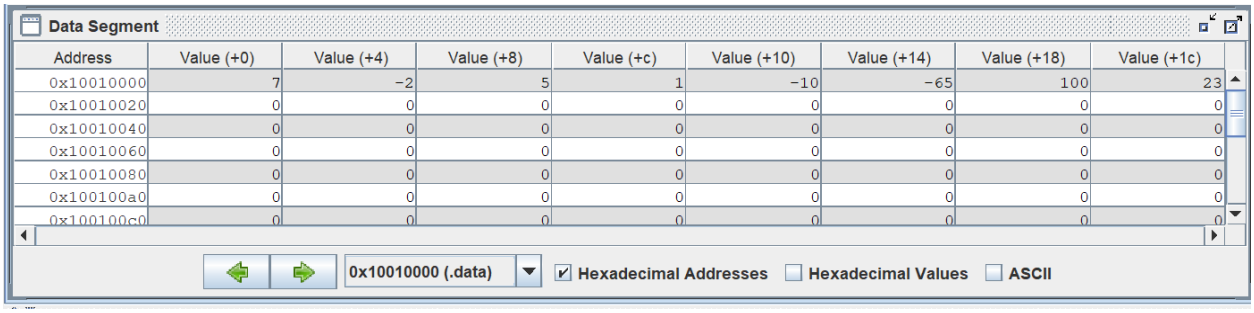
lab 6.asm*

```
19    beq   s0, a1, done   # if i>n
20    lw t1, 0(s0)
21    addi s1, s0, -4 # s1=i-1
22    j     loop
23 done:
24    j     after_sort
25
26 loop:
27    lw t2, 0(s1)
28    bge t1, t2, next
29    blt s1, a0, next
30    addi t0, s1, 4
31    sw t2, 0(t0)
32    addi s1, s1, -4
33    j loop
34 next:
35       addi s1, s1, 4
36       sw t1, 0(s1)
37       addi s0, s0, 4
38       j sort
39 end_sort:
```

Line: 28 Column: 20 ☑ Show Line Numbers

The result of the insertion sort algorithm:

Before sorting:



After sorting: